



TESIS - KI142502

KLASIFIKASI MULTILABEL MOTIF CITRA BATIK MENGUNAKAN BOOSTED RANDOM FERNS DENGAN EKSTRAKSI FITUR HISTOGRAM OF ORIENTED GRADIENT

M. Nur Fuad
5115201022

DOSEN PEMBIMBING
Dr. Eng. Nanik Suciati, S.Kom., M.Kom.

PROGRAM MAGISTER
BIDANG KEAHLIAN KOMPUTASI CERDAS DAN VISUALISASI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2017

[Halaman ini sengaja dikosongkan]

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Komputer (M.Kom.)
di
Institut Teknologi Sepuluh Nopember Surabaya

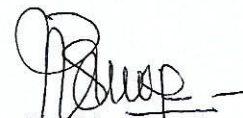
oleh:
M.Nur Fuad
Nrp. 5115201022

Dengan judul :
KLASIFIKASI MULTILABEL MOTIF CITRA BATIK
MENGUNAKAN BOOSTED RANDOM FERNS DENGAN
EKSTRAKSI FITUR HISTOGRAM OF ORIENTED


Tanggal Ujian : 6-7-2017
Periode Wisuda : 2016 Genap

Disetujui oleh:


Dr. Eng. Nanik Suciati, S.Kom, M.Kom
NIP. 197104281994122001


(Pembimbing 1)

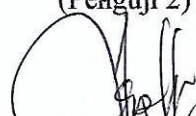
Dr. Eng. Chastine Fatichah, S.Kom, M.Kom
NIP. 197512202001122002


(Penguji 1)

Dini Adni Navastara, S.Kom., M.Sc.
NIP. 198510172015042001


(Penguji 2)

Arya Yudhi Wijaya, S. Kom., M.Kom
NIP. 198409042010121002


(Penguji 3)


Dekan Fakultas Teknologi Informasi,
Dr. Agus Zainal Arifin, S.Kom, M.Kom.
NIP. 197208091993121001

[Halaman ini sengaja dikosongkan]

Klasifikasi Multilabel Motif Citra Batik Menggunakan *Boosted Random Ferns* Dengan Ekstraksi Fitur *Histogram of Oriented Gradient*

Nama Mahasiswa : M. Nur Fuad

NRP : 5115201022

Pembimbing : Dr. Eng.Nanik Suciati, S.Kom.,M.Kom.

ABSTRAK

Terdapat banyak penelitian tentang klasifikasi motif batik, namun hanya mengklasifikasi satu motif batik dalam satu citra. Pada beberapa kasus, dalam satu citra terdapat lebih dari satu motif batik. Sehingga dibutuhkan klasifikasi banyak motif batik dalam satu citra. Masing-masing motif batik memiliki bentuk yang unik meskipun berada dalam satu citra. Deteksi bentuk dapat digunakan untuk mengidentifikasi bentuk motif-motif batik. *Histogram of oriented gradient* (HOG) merupakan ekstraksi bentuk yang telah diaplikasikan pada identifikasi pejalan kaki. Dalam perkembangannya, kombinasi HOG dengan klasifikasi *boosted random ferns* mampu mengidentifikasi beberapa pejalan kaki dalam satu citra. Kemampuan kombinasi metode tersebut dalam mengidentifikasi *multiobject* dalam satu label (pejalan kaki) akan dikembangkan untuk mengidentifikasi *multiobject* dalam multilabel (motif-motif batik).

Berdasarkan hal tersebut disusun sistem klasifikasi *multilabel* motif batik menggunakan *boosted random ferns* dengan ekstraksi fitur HOG. Sistem ini terbagi dalam proses *training* dan *testing*. *Input* proses *training* berupa citra motif parang, kawung, megamendung, nitik, semen dan lunglungan. Fitur HOG diekstraksi dari *input training* dan diproses dengan *boosted random ferns* sehingga menghasilkan *output* berupa enam model klasifikasi motif. *Input* proses *testing* berupa citra *testing* dan enam model klasifikasi motif. Pada proses *testing*, setiap model klasifikasi motif akan mengidentifikasi masing-masing motif dalam citra *testing*. Hasil *output* dari sistem berupa hasil identifikasi dari model-model klasifikasi motif.

Pengujian performa metode menggunakan beberapa skenario pengujian berdasarkan variasi jumlah *subset random ferns*, jumlah *weak classifier* dan iterasi *bootstrapping*. Terdapat empat variasi jumlah *subset random ferns* yakni 5, 10, 15 dan 20 *subset*, empat variasi jumlah *weak classifier* yakni 100, 200, 300 dan 400, serta enam variasi iterasi *bootstrapping* yakni 0, 1, 2, 3, 4, dan 5 iterasi. Sedangkan nilai performa metode dihitung menggunakan *tanimoto distance*. Diperoleh nilai *tanimoto distance* terbaik yakni 0.0130 dari iterasi *bootstrapping* ke-2. Sedangkan tanpa iterasi *bootstrapping*, diperoleh nilai 0.0469 pada jumlah *weak classifier* 200 dengan *subset ferns* 5 dan jumlah *weak classifier* 300 pada subset ferns ke 5 dan 15.

Kata kunci: *histogram of oriented gradient, boosted random ferns, klasifikasi, motif batik, multilabel.*

[Halaman ini sengaja dikosongkan]

Multilable Classification Batik Image Pattern Using Boosted Random Ferns With Histogram of Oriented Gradient Feature Ekstraktion

Student Name : M. Nur Fuad

NRP : 5115201022

Supervisor : Dr. Eng.Nanik Suciati, S.Kom.,M.Kom.

ABSTRACT

There are many studies on the classification of the batik pattern, but only classifies one batik pattern in single image. In some cases, there are single image with more than one batik patterns. So it need a classification of some motif in one image. Each pattern has unique shape even within single image. Detection of shapes can be used to identify the form of batik patterns. Histogram of oriented gradient (HOG) is an shape extraction that has been applied to the identification of the pedestrian. During HOG development, combination of HOG and boosted random *ferns* classification able to identify several pedestrians in single image. Ability of these methods combination in identifying multiobject in one label (pedestrian) will be developed to identify multiobject in multilable (batik patterns).

Based on this, we organized multilable batik patterns classification system using random boosted ferns with HOG feature extraction. System are divided into training and testing process. Input of training process are image of parang, kawung, megamendung, nitik, cement and lunglungan patterns. HOG features extracted from training input and be processed with boosted random ferns to produce output six pattern classification models. Input of testing process are testing image and and six pattern classification models. In testing process, each pattern classification model will identify each patterns in the image testing. Outputs system are identification result from pattern classification models.

Performance testing of proposed method be calculate using multiple test scenarios based on variety numbers of random ferns subsets, number of weak classifier and bootstrapping iterations. There are four variety numbers of random subset ferns ie 5, 10, 15 and 20 subsets, four variety numbers of weak classifier ie 100, 200, 300 and 400, and six variety numbers of bootstrapping iterations ie 0, 1, 2, 3, 4, and 5 iterations. Whereas performance value of methods be calculate using Tanimoto distance. The best tanimoto distance value is 0.0130 from the 2nd iteration bootstrapping. While without bootstrapping iteration, the value of 0.0469 is obtained on the 200 weak classifier with 5 subset ferns and the 300 weak classifier with 5 and 15 subset ferns.

Keywords: *Histogram of oriented gradient, boosted random ferns, classification, batik patterns, multilable.*

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Assalamualaikum warohmatullohi wabarokatuh.

Segala puja dan puji syukur penulis panjatkan kepada Allah SWT atas berkat, rahmat, taufik dan hidayah-Nya sehingga penyusunan dan penulisan Tesis yang berjudul “Klasifikasi Multilabel Motif Citra Batik Menggunakan Boosted Random Ferns Dengan Ekstraksi Fitur Histogram of Oriented Gradient” dapat terselesaikan dengan baik.

Shalawat serta salam semoga tercurah terhadap junjungan Nabi Besar Muhammad saw. Beliau lah yang membimbing umat Islam dari gelapnya kekufuran menuju cahaya Islam yang terang benderang.

Dengan segala kerendahan hati, penulis menyadari keterbatasan pengetahuan yang penulis miliki, karena itu tanpa keterlibatan dan sumbangsih dari berbagai pihak, sulit bagi penulis untuk menyelesaikan tesis ini. Oleh karena itu, dalam kesempatan ini penulis ingin mengucapkan terima kasih tak terhingga kepada :

1. Allah SWT atas limpahan rahmat-Nya sehingga penulis dapat menyelesaikan tesis ini dengan baik.
2. Ibu Siti Mujiati, Bapak M. Musa dan Nenek Khomsatun, selaku orang tua dan Nenek penulis yang senantiasa mendoakan agar penulis selalu diberikan kelancaran dan kemudahan dalam menyelesaikan tesis ini. Serta menjadi motivasi terbesar untuk mendapatkan hasil yang terbaik.
3. Ibu Dr. Eng. Nanik Suciati, S.Kom, M.Kom, selaku dosen pembimbing yang senantiasa membimbing penulis dengan penuh kesabaran dengan tetap memberikan kepercayaan, motivasi, nasehat, perhatian dan semua bantuan sehingga penulis berhasil menyelesaikan tesis ini.
4. Ibu Dr. Eng. Chastine Fatichah, S.Kom, M.Kom., Bapak Arya Yudhi Wijaya, S.Kom., M.Kom., Ibu Bilqis Amaliyah, S.Kom, M.Kom dan Ibu Dini Adni Navastara, S.Kom, M.Sc selaku dosen penguji yang telah memberikan bimbingan, saran, arahan, dan koreksi dalam pengerjaan tesis ini.

5. Bapak Waskitho Wibisono, S.Kom., M.Eng., PhD selaku ketua program pascasarjana Teknik Informatika ITS dan segenap dosen Teknik Informatika yang telah dengan ikhlas memberikan ilmu-ilmu beliau.
6. Mbak Lina, Mas Kunto dan segenap staf Tata Usaha yang telah memberikan segala bantuan dan kemudahan kepada penulis selama menjalani kuliah di Teknik Informatika ITS.
7. Adik-adik dan sepupu penulis, Irsyad Adifa M., Muhammad Sholeh Al Fadhil, Della Zakiyah Awaliyah, Kana Ahsanur Rijal, Nabila Nur Bakkah, Aliza Nur Muhammad, Misbachul Muta'ali, Miqdam Syauqi Nur Muhammad, Nabhan, Nayla Amjad Azzahro, Husein, Mas Rifqi Aghis, Mas Thoifur Ulin Nuha serta seluruh keluarga besar yang senantiasa memberi dorongan moril dan menjadi inspirasi penulis.
8. Rekan-rekan angkatan 2015 Pasca Sarjana Teknik Informatika ITS dan teman-teman penulis yang telah menemani dan memberikan bantuan serta motivasi untuk segera menyelesaikan tesis ini.
9. Juga tidak lupa kepada semua pihak yang belum sempat disebutkan yang telah membantu terselesaikannya tesis ini.

Hanya ucapan terima kasih dan doa tulus penulis berikan atas apa yang telah mereka berikan, semoga apa yang telah mereka lakukan dapat memberikan kebaikan kembali kepada mereka. Aamiin yaa robbal 'aalamiin.

Sebagai manusia biasa, penulis menyadari bahwa tesis ini masih jauh dari kesempurnaan dan memiliki banyak kekurangan. Sehingga dengan segala kerendahan hati, penulis mengharapkan saran dan kritik yang membangun dari pembaca. Wassalamualaikum warohmatullohi wabarokatuh.

Surabaya, 21 Juli 2017

Penulis

DAFTAR ISI

ABSTRAK	v
ABSTRACT	vii
KATA PENGANTAR	ix
DAFTAR ISI	xi
DAFTAR GAMBAR	xiii
DAFTAR TABEL	xv
BAB 1 PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Perumusan Masalah	3
1.3. Tujuan	3
1.4. Manfaat	4
1.5. Kontribusi Penelitian	4
1.6. Batasan Masalah	4
BAB 2 KAJIAN PUSTAKA	5
2.1. Batik	5
2.2. <i>Histogram of Oriented Gradient</i>	5
2.3. <i>Boosted Random Ferns</i>	11
2.3.1. Perhitungan <i>weak classifiers</i> dan <i>strong classifier</i>	13
2.3.2. <i>Bootstrapping</i>	17
2.4. <i>Tanimoto Distance</i>	21
BAB 3 METODOLOGI PENELITIAN	23
3.1. Proses <i>Training</i>	25
3.1.1. Pemilahan dataset	26
3.1.2. Pembangkitan <i>weak classifier</i>	30
3.1.2.1. <i>Histogram of oriented gradient</i>	32
3.1.3. <i>Update bobot weak classifier</i>	35
3.1.4. Pembangkitan <i>strong classifier</i>	36
3.2. Proses <i>Testing</i>	37
3.2.1. Model klasifikasi <i>multilabel</i>	39
3.2.2. Pelabelan data <i>testing</i>	41
3.2.3. <i>Bootstrapping</i>	43

3.2.4. Hitung akurasi	44
3.3. Skenario pengujian	44
BAB 4 HASIL DAN PEMBAHASAN	47
4.1. Implementasi Metode	47
4.2. Hasil Pengujian	48
4.2.1. Hasil pengujian model klasifikasi motif terhadap data training	48
4.2.2. Hasil pengujian model klasifikasi motif terhadap data <i>testing</i>	50
4.2.3. Hasil pengujian <i>bootstrapping</i>	57
4.3. Pembahasan Hasil Pengujian	58
BAB 5 PENUTUP	61
5.1. Kesimpulan	61
5.2. Saran	61
DAFTAR PUSTAKA	63
Lampiran A. Hasil Uji Coba	65
Lampiran B. Tampilan Histogram of Oriented Gradient	73
Biografi Penulis	79

DAFTAR GAMBAR

Gambar 2.1 Motif Parang - Megamendung	6
Gambar 2.2 Motif Parang – Semen.....	6
Gambar 2.3 Matrix I_x	8
Gambar 2.4 Matrix I_y	8
Gambar 2.5 Matrix $\mu = I_x^2 + I_y^2$	8
Gambar 2.6 Matrix $\theta = 90 + (\text{Arctan} I_{xy})$	8
Gambar 2.7 Pemetaan Bin (Tomasi, 2012).....	9
Gambar 2.8 Histogram Bobot Nilai Gradien Dalam Satu <i>Cell</i>	10
Gambar 2.9 Ilustrasi <i>Overlapping Block</i>	10
Gambar 2.10 Alur Algoritma BRF (Villamizar Et Al., 2012).	12
Gambar 2.11 Dataset Positif	13
Gambar 2.12 Dataset Negatif.....	13
Gambar 2.13 <i>Random Ferns</i> Pada Bin Hog (Villamizar Et Al., 2012).	14
Gambar 2.14 Ilustrasi Skala Citra Dalam <i>Testing</i> (Villamizar Et Al., 2012).	18
Gambar 2.15 Ilustrasi Pengambilan Dataset Baru (Villamizar Et Al., 2012).	20
Gambar 2.16 Ilustrasi Distribusi <i>Random Ferns</i> (Villamizar Et Al., 2012).	21
Gambar 3.1 <i>Flowchart</i> Desain Sistem Klasifikasi <i>Multilabel Boosted Random Ferns</i>	23
Gambar 3.2. <i>Flowchart</i> Proses <i>Training</i>	27
Gambar 3.3 <i>Flowchart</i> Pemilahan Dataset Parang	28
Gambar 3.4 <i>Flowchart</i> Penyusunan Model Klasifikasi Motif Parang	33
Gambar 3.5 Bin Orientasi HOG.....	34
Gambar 3.6 <i>Flowchart</i> HOG.....	35
Gambar 3.7 Distribusi <i>Subset Random Ferns</i> (Villamizar Et.Al, 2012).....	36
Gambar 3.8 <i>Flowchart</i> Klasifikasi Dengan <i>Strong Classifier</i>	37
Gambar 3.9 Contoh <i>Ground Truth</i>	38
Gambar 3.10 <i>Flowchart</i> Proses <i>Testing</i>	39
Gambar 3.11 <i>Flowchart</i> Proses Klasifikasi Motif Batik.....	42
Gambar 4.1 Contoh Data <i>Testing</i> Motif Satuan.....	47

Gambar 4.2 Contoh Data Testing Motif Kombinasi	48
Gambar 4.3 Ilustrasi Hasil Deteksi Model Klasifikasi Motif. (A) Data Hasil Deteksi Model Klasifikasi Motif. (B) Data Hasil Deteksi Model Klasifikasi Motif Sesuai <i>Ground Truth</i> . (C) Data Hasil Deteksi Model Klasifikasi Motif Tidak Sesuai <i>Ground Truth</i> . (D) Data <i>Ground Truth</i> Tanpa Hasil Klasifikasi (Tidak Terdeteksi). (E) Data Hasil Deteksi Model Klasifikasi Motif Tanpa <i>Ground Truth</i>	51

DAFTAR TABEL

Tabel 2.1 Contoh Hasil Prediksi	22
Tabel 3.1 Tabel Contoh Hasil Pemilahan Dataset	29
Tabel 4.1 Pengujian Akurasi Jumlah <i>Weak Classifier</i> 100 Pada Data <i>Training</i> ...	49
Tabel 4.2 Pengujian Akurasi Jumlah <i>Weak Classifier</i> 200 Pada Data <i>Training</i> ...	49
Tabel 4.3 Pengujian Akurasi Jumlah <i>Weak Classifier</i> 300 Pada Data <i>Training</i> ...	49
Tabel 4.4 Pengujian Akurasi Jumlah <i>Weak Classifier</i> 400 Pada Data <i>Training</i> ...	50
Tabel 4.5 Contoh Hasil Pelabelan Deteksi Motif.....	52
Tabel 4.6 Pengujian Jumlah <i>Weak Classifier</i> 100 Pada Data <i>Testing</i>	53
Tabel 4.7 Pengujian Jumlah <i>Weak Classifier</i> 200 Pada Data <i>Testing</i>	53
Tabel 4.8 Pengujian Jumlah <i>Weak Classifier</i> 300 Pada Data <i>Testing</i>	53
Tabel 4.9 Pengujian Jumlah <i>Weak Classifier</i> 400 Pada Data <i>Testing</i>	53
Tabel 4.10 Pengujian Terhadap Data <i>Testing</i> (5 <i>Ferns</i> Dengan Jumlah <i>Weak Classifier</i> 100).....	54
Tabel 4.11 Lanjutan Tabel 4.10	55
Tabel 4.12 Lanjutan Tabel 4.11	56
Tabel 4.13 Pengujian <i>Boostrapping</i>	57

[Halaman ini sengaja dikosongkan]

BAB 1

PENDAHULUAN

Pada bab ini dijelaskan mengenai beberapa hal dasar dalam pembuatan penelitian yang meliputi latar belakang, perumusan masalah, tujuan, manfaat, kontribusi penelitian, dan batasan masalah.

1.1. Latar Belakang

Dalam batik terkandung teknik, simbolisme dan budaya yang meliputi kehidupan masyarakat Indonesia dari awal sampai akhir. Mulai dari selimut batik untuk bayi yang dihiasi simbol-simbol yang dirancang untuk membawa keberuntungan bagi bayi sampai kain batik yang digunakan untuk penguburan jenazah. Batik juga dipakai sebagai pakaian sehari-hari seperti ketika sedang bekerja dan bersekolah, bahkan batik dipakai untuk menghadiri pernikahan dan acara resmi (www.unesco.org).

Kemudian pada tanggal 2 Oktober 2009, UNESCO menetapkan batik ke dalam Daftar Representatif Warisan Kemanusiaan untuk Budaya Non Bendawi (*Representative List of the Intangible Cultural Heritage of Humanity*). Sejak saat itu semakin banyak penelitian tentang batik dan klasifikasi motif batik, termasuk penelitian dalam pengolahan citra digital. Pada pengolahan citra digital, citra batik dapat diteliti ciri tekstur, warna maupun bentuk dari motif batik. Dari ciri motif batik yang diteliti dapat dilakukan klasifikasi untuk menentukan jenis motif batik.

Pratikaningtyas et al. (2010) melakukan penelitian klasifikasi motif batik menggunakan metode transformasi paket *wavelet* dengan metode klasifikasi jarak *euclidean*. Kemudian Riesmala et al. (2012) menggunakan ekstraksi fitur statistik ciri orde satu dan ciri orde dua dengan klasifikasi *k-NN* untuk klasifikasi batik berdasarkan struktur dan warna motifnya. Setelah itu terdapat penelitian mengenai klasifikasi kansei multi label pada citra batik oleh Nilogiri et al. (2012) yang menggunakan konsep *multilabel* terhadap variasi kansei berdasarkan warna, tekstur dan bentuknya. Lalu Rangkuti (2013) menggunakan *wavelet transform* jenis *daubechies* 2 level 2, untuk memproses ciri tekstur yang terdiri dari standard deviasi, mean dan energi sebagai variabel *input* dengan metode klasifikasi *fuzzy*

neural network. Selanjutnya Kurniawardhani et al. (2014) menggunakan *rotation complete robust local binary pattern magnitude* (rotCRLBP_M) untuk memperoleh fitur tekstur yang invarian terhadap rotasi tapi tetap mempertahankan ciri *magnitude*, dengan metode klasifikasi *probabilistic neural network*. Kemudian Dhian E K R & Nugraha (2016) menggunakan transformasi *wavelet daubechies* level 4 (DB-4) dan *gabor* sebagai ekstraksi fitur dengan metode klasifikasi *K-NN* untuk mendapatkan hasil akurasi yang lebih optimal dari penelitian sebelumnya. Selanjutnya Sulistyo A.S. (2016) menerapkan metode *learning vector quantization* dalam perangkat android untuk mendeteksi pola motif batik dengan fitur dari metode frekuensi tepi yang merupakan hasil deteksi tepi *canny*.

Dalam penelitian-penelitian tersebut terdapat satu kelemahan yang sama, yakni tidak dapat menampilkan hasil klasifikasi terhadap lebih dari satu variasi motif batik dalam satu citra batik. Penelitian-penelitian tersebut fokus pada identifikasi satu jenis motif batik dalam satu citra batik. Padahal dalam beberapa kasus, terdapat lebih dari satu jenis motif batik dalam satu citra batik. Oleh karena itu dibutuhkan suatu sistem yang mampu mengidentifikasi lebih dari satu motif batik dalam satu citra batik.

Salah satu cara untuk mengenali lebih dari satu motif batik dalam satu citra adalah dengan melakukan identifikasi bentuk, karena bentuk setiap motif batik memiliki keunikan bentuk tersendiri meskipun berada dalam satu citra. Dengan identifikasi bentuk, terdapat kemungkinan untuk mengidentifikasi lebih dari satu motif dalam satu citra. Salah satu ekstraksi bentuk yang dapat digunakan yakni *histogram of oriented gradient* (HOG). HOG terkenal digunakan untuk melakukan deteksi pejalan kaki (Dalal & Triggs, 2005). HOG juga dapat digunakan dalam deteksi bentuk pada penelitian *image retrieval* seperti yang dilakukan oleh Asiri, et al. (2015) dan Randa et al.(2016).

Dalam perkembangan HOG, HOG yang dikombinasikan dengan klasifikasi *boosted random ferns* (Villamizar et al, 2012) dapat digunakan untuk mendeteksi beberapa pejalan kaki sekaligus. Hanya saja meskipun mampu

mendeteksi beberapa objek sekaligus pada satu citra, namun masih dalam satu label yang sama. Hal ini seperti jika pada suatu citra terdapat objek mobil, motor dan pejalan kaki. Maka hanya satu jenis objek saja yang akan terdeteksi, seperti mobil saja, motor saja atau pejalan kaki saja. Untuk mampu mendeteksi objek-objek dengan jenis label yang berbeda dalam satu citra, diperlukan modifikasi pada sistem ekstraksi fitur HOG dengan klasifikasi *boosted random ferns*.

Berdasarkan uraian tersebut penulis berusaha untuk mengusulkan sebuah sistem klasifikasi yang dapat mendeteksi beberapa motif batik dalam satu citra. Sistem ini menggunakan ekstraksi fitur bentuk HOG untuk mendapatkan ciri bentuk motif batik dan *boosted random ferns* sebagai metode klasifikasi motif batik. *Boosted random ferns* menggunakan dataset positif dan negatif sebagai data *training*. Susunan dataset ini dimodifikasi supaya dapat digunakan untuk klasifikasi multilabel, karena dataset *boosted random ferns* original hanya didesain untuk *single* label atau berupa klasifikasi biner yang hanya membedakan objek dan *background*.

1.2. Perumusan Masalah

Rumusan masalah yang diangkat dalam penelitian ini adalah sebagai berikut.

1. Bagaimana cara mengekstraksi fitur HOG pada citra batik ?
2. Bagaimana cara melakukan klasifikasi multilabel motif citra batik menggunakan *boosted random ferns* ?
3. Bagaimana cara mengetahui performa dari klasifikasi multilabel motif citra batik menggunakan *boosted random ferns* ?

1.3. Tujuan

Tujuan penelitian ini adalah melakukan klasifikasi multilabel pada motif citra batik menggunakan sistem klasifikasi yang dapat mengenali berbagai jenis motif pada citra batik.

1.4. Manfaat

Penelitian ini diharapkan dapat membantu masyarakat yang awam tentang batik dalam mengenali berbagai jenis motif batik.

1.5. Kontribusi Penelitian

Kontribusi penelitian ini adalah menghasilkan sebuah sistem klasifikasi *multilabel* menggunakan *boosted random ferns* yang dapat melakukan klasifikasi berbagai motif batik pada citra batik.

1.6. Batasan Masalah

Batasan masalah pada penelitian ini adalah:

1. Dataset *training* yang digunakan adalah dataset citra batik klasik dengan motif parang (40 citra), kawung (40 citra), lunglungan (30 citra), megamendung (30 citra), nitik (50 citra), semen (30 citra), serta dataset *background* (120 citra) yang tidak termasuk motif manapun.
2. Dataset testing terdiri dari 64 citra batik yang terbagi menjadi 42 citra motif satuan dan 22 citra motif kombinasi. Citra motif satuan terdiri dari 6 citra parang, 9 citra kawung, 6 citra lunglungan, 9 citra megamendung, 6 citra semen dan 6 citra nitik. Citra motif kombinasi terdiri dari 2 citra kombinasi parang-kawung-lunglungan, 4 citra kombinasi parang-megamendung, 7 citra kombinasi parang-nitik, 3 citra kombinasi parang-semen, 3 citra kombinasi kawung-nitik dan 3 citra kombinasi semen lunglungan.

BAB 2

KAJIAN PUSTAKA

Pada bab ini dijelaskan tentang pustaka yang terkait dengan landasan penelitian. Pustaka yang terkait adalah tentang batik, histogram of oriented gradient dan boosted random *ferns*.

2.1. Batik

Batik merupakan warisan budaya turun temurun dengan nilai estestika tinggi dan filosofi yang mendalam. Pada awalnya, batik hanya terbatas dipakai oleh keluarga kerajaan sehingga motif batik sekaligus menunjukkan status pemakainya sedangkan perbedaan corak menunjukkan asal keluarga tertentu. Kemudian dengan berkembangnya zaman, masyarakat memakai motif batik corak kraton tanpa memperhitungkan etika pemakaian (Sulistyo A.S., 2016). Batik memiliki berbagai macam jenis corak atau pola Batik. Pola-pola tersebut disusun secara berulang untuk menggambarkan motif dasar pada suatu kain secara keseluruhan. Perulangan motif pada Batik dapat disusun secara teratur maupun tidak teratur (Kurniawardhani et al., 2014).

Batik dapat digunakan dalam berbagai kegiatan dan keadaan karena memiliki banyak variasi motif. Berbagai macam variasi motif ini dikarenakan batik dibuat oleh daerah-daerah yang berbeda. Masing-masing motif memiliki makna dan menjadi representasi khas masing-masing daerah (Riesmala et al., 2012). Seperti motif megamendung dari Cirebon, parang dan kawung dari Solo dan Jogja, serta berbagai motif dari Pekalongan (Jlamprang, Liong dan lainnya). Awalnya dalam satu kain batik hanya terdapat satu motif batik. Namun pada perkembangannya dalam satu kain bisa terdapat lebih dari satu motif batik. Contoh batik dengan motif lebih dari satu motif dalam satu kain ditampilkan dalam Gambar 2.1 dan Gambar 2.2.

2.2. *Histogram of Oriented Gradient*

Histogram of oriented gradient (HOG) merupakan deskriptor fitur berbasis window yang digunakan pada *computer vision* dan *image processing*

untuk mendeteksi objek. Deteksi objek dilakukan HOG dengan cara melakukan ekstraksi fitur bentuk dari objek pada citra. Fitur-fitur HOG diperoleh dengan menghitung nilai orientasi gradien dengan bobot nilai gradien pada *region* tertentu dalam suatu citra. *Region* ini disebut sebagai *cell*, dimana *cell* didapatkan dengan cara membagi citra ke dalam *region-region* kecil dengan ukuran sama.



Gambar 2.1 Motif parang - megamendung



Gambar 2.2 Motif parang – semen

Pada setiap *cell* dihitung sebuah histogram 1 dimensi dari nilai bobot bin orientasi gradien. Selanjutnya beberapa *cell* dikelompokkan ke dalam *region* yang lebih besar (*block*) dan histogram dari semua *cell* dalam satu *block* dikombinasi dan dinormalisasi ke dalam satu vektor *block*. *Block-block* yang telah dinormalisasi tersebut merupakan deskriptor fitur bentuk dari HOG (Dalal & Triggs, 2005).

Berikut langkah-langkah perhitungan ekstraksi fitur HOG :

1. Menghitung nilai gradien dan orientasi gradien citra

Untuk mendapatkan nilai gradien, diperlukan nilai tepi dari citra. Salah satu cara memperoleh nilai tepi citra adalah dengan menggunakan magnitude, yaitu dengan menggabungkan nilai magnitude terhadap sumbu x dengan magnitude terhadap sumbu y. Misalkan untuk mendapatkan nilai magnitude terdapat citra *grayscale* (I) dalam bentuk image double, maka nilai magnitude terhadap sumbu x (I_x) diperoleh menggunakan filtering citra I terhadap kernel D_x , sedangkan nilai magnitude terhadap sumbu y (I_y) diperoleh dengan

melakukan filtering terhadap kernel D_y . Kernel D_x yang digunakan berupa matrik $\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$ sedangkan kernel D_y berupa matrik $\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$. Hasil nilai I_x dan I_y ini memiliki range antara -1 sampai 1. Perhitungan untuk memperoleh I_x dan I_y ditampilkan dalam persamaan 2.1. Setelah nilai I_x dan I_y diperoleh, maka nilai magnitude (μ) dihitung dengan melakukan akar kuadrat terhadap penjumlahan dari I_x^2 dan I_y^2 seperti dalam persamaan 2.2. Selanjutnya nilai μ digunakan sebagai nilai gradien dari citra.

$$I_x = I * D_x, I_y = I * D_y \quad (2.1)$$

$$\mu = \sqrt{I_x^2 + I_y^2} \quad (2.2)$$

Selanjutnya dilakukan perhitungan nilai orientasi gradien (θ). Dalam orientasi gradien (θ) terdapat dua jenis tipe orientasi yakni orientasi *signed* (antara 0° sampai 360°) dan *unsigned* (antara 0° sampai 180°). Dalam HOG digunakan orientasi *unsigned* sehingga nilai orientasi hanya berkisar antara 0° sampai 180°. Oleh karena itu, diperlukan penambahan nilai 90 terhadap perhitungan θ yang didapatkan dengan melakukan perkalian arctan terhadap hasil bagi dari I_x dibagi I_y . Hal ini untuk mencegah terdapat nilai derajat negatif (dengan membatasi nilai derajat minimum 0°) dan membatasi nilai derajat maksimum menjadi 180°. Persamaan untuk mendapatkan θ ditampilkan dalam persamaan 2.3.

$$\theta = 90 + (\arctan\left(\frac{I_x}{I_y}\right)) \quad (2.3)$$

Contoh matrik magnitude I_x dan I_y ditampilkan pada Gambar 2.3 dan Gambar 2.4. Kemudian matrik μ dan matrik θ dari I_x dan I_y ditampilkan pada Gambar 2.5 dan Gambar 2.6.

2. Menghitung nilai bin orientasi gradien dalam tiap *cell*

Setelah nilai gradien (μ) dan orientasi gradien (θ) didapatkan, selanjutnya citra dibagi menjadi *region-region* berukuran CxC piksel yang saling tidak *overlapping*. *Region* ini disebut dengan *cell* dengan ukuran C = 8. Di dalam *cell* akan didapatkan vektor yang berisi nilai histogram 1 dimensi dari nilai bobot bin orientasi gradien. Histogram ini menggunakan nilai bin 9 buah (0 sampai 8) yang digunakan untuk melakukan *voting* terhadap nilai θ dan μ .

-1	-1	0	1	0	-1	0	0
-1	1	0	1	0	0	0	1
0	0	0	0	0	0	0	1
-1	-1	0	1	0	0	0	1
-1	-1	0	1	0	-1	0	0
0	0	0	0	0	0	0	1
0	-1	0	1	0	-1	0	0
-1	-1	0	1	0	0	0	1

Gambar 2.3 Matrix I_x

1	0	0	-1	0	0	0	-1
0	0	0	0		0	0	0
-1	0	0	1	0	0	0	-1
1	0	0	-1	0	0	0	0
1	0	0	0	0	0	0	0
-1	0	0	1	0	0	0	0
0	1	0	0	0	1	0	0
1	1	0	0	0	1	0	1

Gambar 2.4 Matrix I_y

1.41	1	0	1.41	0	1	0	1
1	1	0	1	0	0	0	1
1	0	0	1	0	0		1.41
1.41	1	0	1.41	0		0	1
1.41	1	0	1	0	1	0	0
1	0	0	1	0	0	0	1
0	1.41	0	1	0	1.41	0	0
1.41	1.41	0	1	0	1	0	1.41

Gambar 2.5 Matrix $\mu = \sqrt{I_x^2 + I_y^2}$

45	0	0	45	0	0	0	90
0	0	0	180	0	0	0	180
90	0		90	0	0	0	135
45		0	45	0	0	0	180
45	0	0	180	0	0	0	0
90	0	0	90	0	0	0	180
0	45	0	180	0	45	0	0
45	45	0	180	0	90	0	135

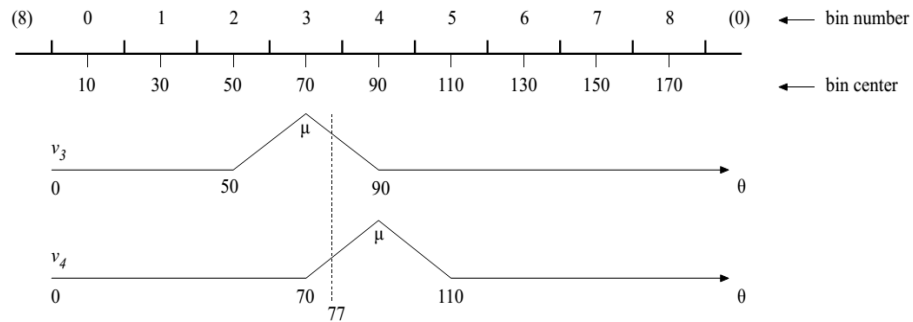
Gambar 2.6 Matrix $\theta = 90 + (\arctan(\frac{I_x}{I_y}))$

Masing-masing nilai bin memiliki lebar w dan nilai tengah c , dimana $w = \frac{180}{B}$ (dengan B adalah jumlah nilai bin sehingga $w = 20$) sedangkan c berada di tengah-tengah masing-masing w . Pemetaan nilai bin dan nilai tengah c dapat dilihat pada Gambar 2.7.

Dalam melakukan *voting* (v), nilai θ akan dihitung terhadap bin ke j (bin dengan $c > \theta$) dan bin ke $j+1$ (bin dengan $c \leq \theta$). Nilai v dihitung menggunakan persamaan 2.4 dan persamaan 2.5. Misalkan θ memiliki nilai 77° , maka θ dihitung terhadap bin ke $j=3$ dan bin ke $j+1=4$ seperti ilustrasi pada Gambar 2.7.

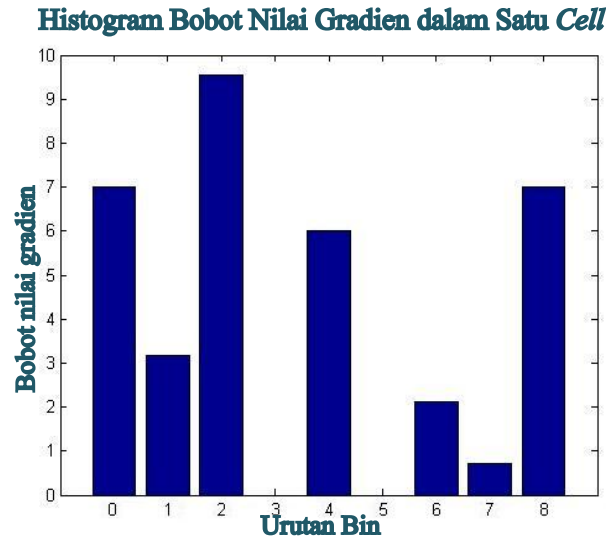
$$v_j = \mu \frac{c_{j+1} - \theta}{w} \text{ untuk bin ke } j = \left(\frac{\theta}{w} - \frac{1}{2}\right) \bmod B \quad (2.4)$$

$$v_{j+1} = \mu \frac{\theta - c_j}{w} \text{ untuk bin ke } j = j + 1 \bmod B \quad (2.5)$$



Gambar 2.7 Pemetaan bin (Tomasi, 2012)

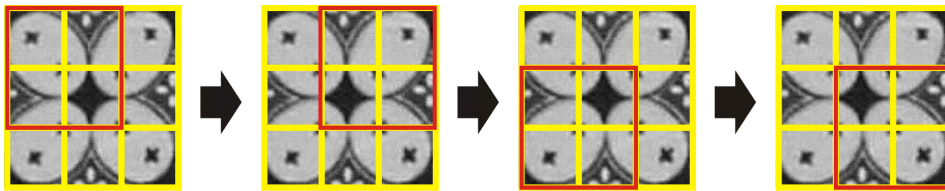
Pada Gambar 2.7 terlihat bahwa 77° berada antara center bin 3 (70°) dan bin 4 (90°). Misal diberikan $\mu = 1$ pada contoh $\theta = 77^\circ$, maka hasil dari voting $v_3 = 1 \frac{90 - 77}{20} = 0.65$ sedangkan voting $v_4 = 1 \frac{77 - 70}{20} = 0.35$. Berdasarkan perhitungan tersebut bobot 0.65 ditambahkan ke dalam bin 3 sedangkan bobot 0.35 ditambahkan ke dalam bin 4. Kemudian setelah voting selesai, maka hasilnya akan berupa vektor histogram 1 dimensi dari nilai bobot bin orientasi gradien. Contoh hasil histogram dari matrik pada Gambar 2.5 dan 2.6 ditampilkan pada Gambar 2.8.



Gambar 2.8 Histogram bobot nilai gradien dalam satu *cell*

3. Melakukan normalisasi histogram tiap *cell* dalam *block*

Langkah terakhir dari HOG adalah melakukan normalisasi tiap vektor histogram *cell*. Normalisasi ini dilakukan untuk menyeragamkan lokal kontras sehingga didapatkan performa yang baik. Normalisasi dilakukan dalam *region* yang lebih besar (*block*) berukuran 2×2 *cell* atau $2C \times 2C$ piksel yang membagi citra secara *overlapping*. Tiap *block* akan melakukan *overlapping* pada *cell* selanjutnya. Ilustrasi *overlapping* ditampilkan pada Gambar 2.9. Pada Gambar 2.9 kotak kecil berwarna kuning adalah *cell* berukuran $C \times C$ sedangkan kotak merah adalah *block* berukuran $2C \times 2C$. Selanjutnya tiap 4 *cell* dalam satu *block* dinormalisasi menggunakan persamaan $L2 - norm$ (persamaan 2.6).



Gambar 2.9 Ilustrasi *Overlapping Block*.

$$b = \frac{b}{\sqrt{\|b\|^2 + \epsilon}} \quad (2.6)$$

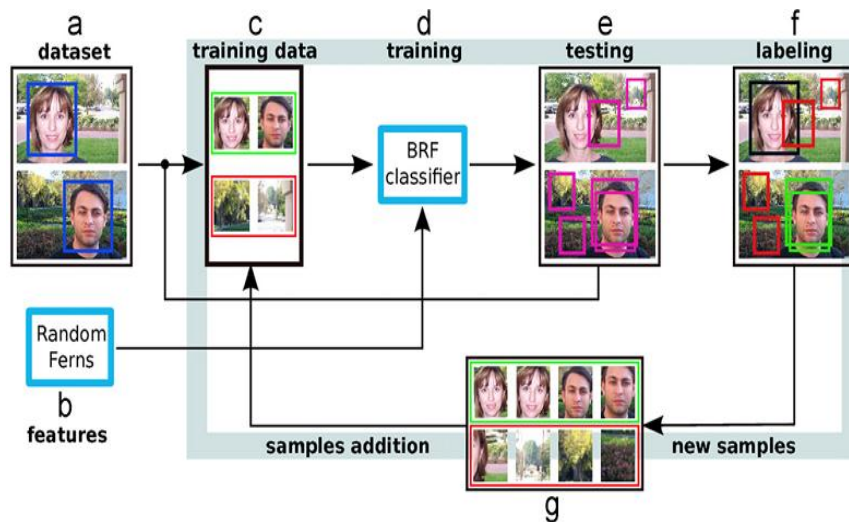
Pada persamaan 2.6, b merupakan fitur *block* dari 4 vektor histogram *cell* dan ϵ merupakan bilangan positif konstan yang bernilai kecil untuk mencegah pembagian dengan 0. Sebelum melakukan normalisasi, 4 vektor *cell* dalam *block* digabung sehingga vektor *block* memiliki panjang 36x1 vektor. Selanjutnya dicari nilai akar dari penjumlahan kuadrat nilai vektor *block* ditambah nilai ϵ . Hasil nilai akar tersebut kemudian digunakan untuk membagi masing-masing nilai dari vektor *block* original sehingga didapatkan nilai vektor *block* yang telah dinormalisasi. Selanjutnya semua nilai vektor *block* yang telah dinormalisasi digabung menjadi vektor HOG. Vektor HOG tersebut memiliki panjang sama dengan jumlah *block* dikali panjang vektor dari satu *block*. Misalkan terdapat citra seperti pada Gambar 2.9, maka jumlah *block* dalam satu citra adalah 2x2 *block* sedangkan panjang satu vektor *block* adalah 36x1 vektor, sehingga panjang vektor HOG pada Gambar 2.9 adalah $36 \times 4 = 144$.

2.3. *Boosted Random Ferns*

Boosted random ferns (BRF) merupakan algoritma yang berdasar pada kombinasi *boosted* dari *random ferns* terhadap *local* HOGs, yang dihitung dalam proses preprosesing (Villamizar et al., 2010). Algoritma BRF memiliki tiga komponen utama yakni *binary descriptor*, *boosting* dan *bootstrapping*. Dalam *binary descriptor* algoritma ini terinspirasi dari *descriptor ferns* sehingga menggunakan representasi *binary* dari *region* citra yang diperoleh berdasarkan sebagian kecil perbandingan piksel yang berpasangan. Hasil dari *binary descriptor* selanjutnya digunakan sebagai *weak classifier*.

Kemudian dalam *boosting* digunakan *real adaboost* untuk mendapatkan *strong classifier* dari kombinasi linear *weak classifiers*. Selanjutnya iterasi *boosting* dilakukan menggunakan strategi *bootstrapping*. Dalam setiap iterasi dengan strategi *bootstrapping*, *strong classifier* dievaluasi terhadap dataset *training* dan data *misclassified* ditambahkan ke dalam dataset yang akan digunakan dalam *training boosting* berikutnya. Hasil proses *bootstrapping* merupakan *classifier* yang lebih sesuai terhadap sampel dan memiliki *global error*

rate yang lebih kecil. Ilustrasi alur sistem dari BRF ditampilkan pada Gambar 2.10.



Gambar 2.10 Alur algoritma BRF (Villamizar et al., 2012).

Pada Gambar 2.10, dataset yang digunakan dibagi menjadi dua label yakni positif untuk objek dan negatif untuk *background* (bagian a). Selanjutnya dilakukan perhitungan fitur *random ferns* yang didefinisikan sebagai perbandingan dari bin-bin pada *local HOG* (bagian b). Perhitungan *random ferns* tersebut dilakukan secara konstan dalam seluruh proses learning dengan data *training* berupa data positif dan negatif yang ditambah dengan data *ground truth* (bagian c). Kemudian digunakan *real adaboost* untuk membangun klasifikasi BFR (bagian d).

Selanjutnya proses *training* diiterasi sesuai dengan sistem *bootstrap* di mana klasifikasi BRF diujikan terhadap seluruh data *training* (bagian e). Karena *ground truth* sudah tersedia, maka hasil deteksi dapat diberi label sebagai true positif (hijau), false positif (merah) dan tidak terdeteksi (hitam) (bagian f). Terakhir data-data yang tidak terdeteksi secara tepat akan dimasukkan ke dalam data yang sesuai, seperti data false positif yang tidak sesuai *ground truth* akan dimasukkan ke dalam dataset negatif sedangkan data *ground truth* yang tidak terdeteksi akan dimasukkan ke dalam data positif (bagian g) (Villamizar et al.,

2012). Hal tersebut dilakukan untuk memperbaiki dataset pada tahap *training* berikutnya. Contoh dataset positif dan negatif dapat dilihat pada Gambar 2.11 dan 2.12.



Gambar 2.11 Dataset positif
(Villamizar et al., 2012)



Gambar 2.12 Dataset negatif
(Villamizar et al., 2012)

Penyusunan algoritma BRF dibagi menjadi dua tahap, yakni pertama perhitungan *weak classifiers* dan *strong classifier*, dan yang kedua *boostapping* untuk mendapatkan *classifier* yang lebih sesuai terhadap sampel dan memiliki *global error rate* yang lebih kecil.

2.3.1. Perhitungan *weak classifiers* dan *strong classifier*

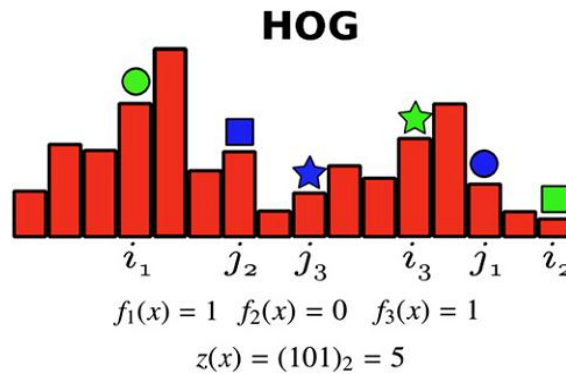
Dalam perhitungan *weak classifiers* dan *strong classifier*, dibutuhkan perhitungan *random ferns* terhadap *local* HOGs. Langkah pertama yang dilakukan yakni memilih posisi image dari piksel secara random (oleh karena itu disebut *random ferns*). Selanjutnya dilakukan operasi *local* HOG pada bagian kecil dari citra yang telah dipilih dan dilakukan perhitungan *random ferns* terhadap bin pada *local* HOG. *Random ferns* dihitung terhadap *local* HOG dengan tujuan untuk meningkatkan *robustness* terhadap pencahayaan dan perubahan *intra-class*.

Misalkan terdapat HOG_x , yang berarti HOG berdasarkan patch citra yang berpusat pada titik x , maka didefinisikan *subset random ferns* $F = [f_{1(x)}, \dots,$

$f_{R(x)}]^T$ sebagai suatu R -dimensional binary vektor dimana nilai $f_{k(x)}$ adalah nilai *fern* yang didapatkan menggunakan menggunakan persamaan 2.7. *Subset random ferns* sendiri merupakan suatu region dalam citra yang berpusat pada posisi tertentu yang mewakili fitur *random ferns* citra tersebut.

$$f_{k(x)} = \begin{cases} 1 & \text{HOG}_x(i) > \text{HOG}_x(j) \\ 0 & \text{HOG}_x(i) \leq \text{HOG}_x(j) \end{cases} \quad (2.7)$$

Pada persamaan 2.7, i dan j merupakan nilai bin dari histogram yang dipilih selama proses *training* secara random *Subset random ferns* F pada posisi x didefinisikan dengan $F: x \rightarrow z$ di mana z merupakan nilai gabungan nilai biner *ferns* f sejumlah K ($z = 1, 2, \dots, K$). Contoh perhitungan 3-dimensional *random ferns* ditampilkan dalam Gambar 2.13. Pada Gambar 2.13 terdapat 2 jenis bin HOG (i dan j) yang saling dibandingkan. Kemudian diperoleh 3-dimensional *random ferns* dari Gambar 2.13 yang menampilkan nilai biner *ferns* $f_{1(x)} = 1$ berdasarkan perbandingan bin HOG $i_1 > j_1$, $f_{2(x)} = 0$ berdasarkan perbandingan bin HOG $i_2 \leq j_2$ dan $f_{3(x)} = 1$ berdasarkan perbandingan bin HOG $i_3 > j_3$. Nilai-nilai biner $f_{1(x)}$, $f_{2(x)}$ dan $f_{3(x)}$ digabungkan ke dalam $z(x) = (101)_2 = 5$. Selanjutnya nilai desimal dari $z(x)$ akan digunakan sebagai fitur *subset random ferns* F pada posisi x (Villamizar et al., 2012). Kemudian semua F dikumpulkan dalam kumpulan *subset random ferns* $\theta = \{F_1, \dots, F_M\}$ di mana M adalah jumlah dari *subset random ferns* yang digunakan.



Gambar 2.13 *Random ferns* pada bin HOG (Villamizar et al., 2012).

Setelah itu *weak classifiers* dibentuk berdasarkan klasifikasi *random ferns* yang dibangun dengan model *real adaboost (boosting)*. Dalam model *boosting*, masing-masing *subset random ferns* $F^t \in \theta$ disusun dalam *weak classifier*nya beserta posisi masing-masing (posisi g^t). *Weak classifier-weak classifier* selanjutnya dikombinasi untuk menghasilkan *strong classifier* H_{C_j} sehingga dapat membedakan kelas C_j dengan background (B) dengan lebih akurat. *Strong classifier* H_{C_j} dihitung menggunakan algoritma *real adaboost* yang menggunakan *weak classifier* sejumlah T , di mana pada setiap iterasi t (bagian dari iterasi T) dilakukan *update* terhadap bobot *weak classifier* berdasarkan kesalahan klasifikasi *weak classifier* dari iterasi sebelumnya.

Persamaan untuk menghitung *strong classifier* H_{C_j} dari didefinisikan dalam persamaan 2.8.

$$H_{C_j}(x) = \sum_{t=1}^T h_{C_j}^t(x) > \beta_{C_j} \quad (2.8)$$

Pada persamaan 2.8, x adalah citra sampel, β_{C_j} adalah *classifier threshold* yang biasanya bernilai 0 dan $h_{C_j}^t$ adalah *weak classifier* yang dihitung menggunakan persamaan 2.9.

$$h_{C_j}^t(x) = \frac{1}{2} \log \frac{P(F^t|C_j, g^t, z^t(x)=k) + \epsilon}{P(F^t|B, g^t, z^t(x)=k) + \epsilon} \quad (2.9)$$

Pada persamaan 2.9, nilai $k = 1, 2, \dots, K$ yang merupakan jumlah biner dalam z sedangkan ϵ berupa *smoothing factor* untuk mencegah pembagian dengan nilai 0. Kemudian pada setiap iterasi *boosting* (t), probabilitas $P(F^t|C_j, g^t, z^t)$ dan $P(F^t|B, g^t, z^t)$ dihitung berdasarkan distribusi bobot (W) terhadap semua citra *training*. Persamaan untuk menghitung W ditampilkan dalam persamaan 2.10 dan 2.11.

$$P(F^t|C_j, g^t, z^t(x) = k) = \sum_{\substack{i: z^t(x_i) = k \\ y_i = +1}} W^t(x_i) \quad (2.10)$$

$$P(F^t|B, g^t, z^t(x) = k) = \sum_{\substack{i: z^t(x_i) = k \\ y_i = -1}} W^t(x_i) \quad (2.11)$$

Pada persamaan 2.10 dan 2.11, nilai $i = 1, 2, \dots, N$ merupakan jumlah suatu set sampel *training*. Untuk memilih *weak classifier* $h_{C_j}^t$ yang paling diskriminatif pada setiap iterasi t maka digunakan perhitungan *bhattacharya distance* Q_t . Perhitungan Q_t dihitung menggunakan persamaan 2.12.

$$Q^t = 2 \sum_{k=1}^K \sqrt{P(F^t|C_j, g^t, z^t(x) = k)P(F^t|B, g^t, z^t(x) = k)} \quad (2.12)$$

Kemudian setelah *weak classifier* didapatkan, hasil klasifikasi digunakan untuk melakukan *update* distribusi bobot W^t . Persamaan untuk menghitung perubahan bobot ditampilkan pada persamaan 2.13.

$$W^{t+1}(x_i) = \frac{W^t(x_i) \exp(y_i h_{C_j}^t(x_i))}{\sum_{i=1}^N W^t(x_i) \exp(y_i h_{C_j}^t(x_i))} \quad (2.13)$$

Di mana W^{t+1} merupakan perubahan bobot dan y_i adalah indikasi sampel adalah objek atau *background*. Keseluruhan proses perhitungan *weak classifiers* dan *strong classifier* ditampilkan dalam *pseudocode* Algoritma 1 (Villamizar et al, 2017). Pada penelitian Villamizar (2012), secara empiris digunakan *weak classifier* sejumlah $T = 300$, *subset random ferns* sejumlah $M = 15$ dan fitur binary sejumlah $R = 7$ pada setiap *subset fern*. Ukuran kumpulan *subset random ferns* dan jumlah fitur dalam setiap *subset fern* menentukan tingkat *recognition* dan beban komputasi.

Algoritma 1. Boosted Random Ferns

Input : dataset *training* untuk kelas C_j , didengan label $\{+1, -1\}$ untuk mengindikasikan sampel termasuk pada kelas C_j atau *background* B

Output : Model klasifikasi object H_{C_j}

- $\theta = \{F_1, \dots, F_M\}$: merupakan kumpulan *random ferns*
- T, β_{C_j} : merupakan jumlah *weak classifiers* yang digunakan, dan *classifier threshold*.

1. Inisiasi distribusi weights terhadap sampel *training* ke $W^1(x_i) = \frac{1}{N}$, di mana $i = 1, 2, \dots, N$.
2. Buat *list* posisi g^t sejumlah L *subset ferns* yang digunakan.
3. Buat sejumlah R kumpulan θ yang berisi F dengan masing-masing pasangan bin HOG yang dicocokkan.
4. **For** $t = 1$ sampai T **do**
5. **For** $l = 1$ sampai L **do**
6. **For** $r = 1$ sampai R **do**
7. Gunakan distribusi weight saat ini (W^t) untuk mendapatkan random fern $F^t \in \theta$ dan posisi citra g^t yang memiliki nilai *Bhattacharyya distace* Q_t terkecil.
8. **End**
9. **End**
10. Gunakan F^t dan g^t untuk menghitung *weak classifier* $h_{C_j}^t$.
11. *Update* distribusi weight W^t .
12. **End**
13. Susun *strong classifier* dari keseluruhan *weak classifier-weak classifier* t dan setiap kombinasi posisi citra g^t dengan pasangan bin HOG yang terpilih oleh *Bhattacharyya distace* pada setiap iterasi t disimpan untuk testing.

2.3.2. Bootstrapping

Pada bagian sebelumnya telah dijelaskan bagaimana mendapatkan *weak classifiers* dan *strong classifier*. Selanjutnya *classifier* yang telah didapatkan diintegrasikan terhadap *framework bootstrapping* yang merubah dataset *training* secara sekuensial untuk digunakan dalam *training classifier* berikutnya. Berikut ini skema *bootstrapping* yang digunakan dalam BRF (Villamizar et al., 2012):

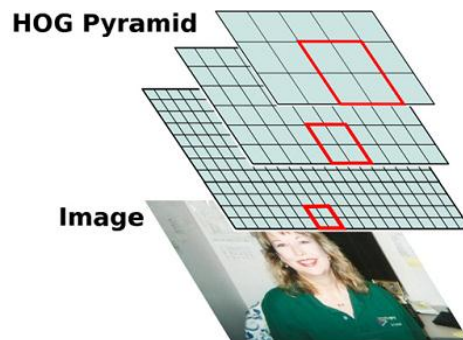
2.3.2.1. Inisiasi dataset *training*

Diasumsikan terdapat label untuk data positif (objek) dan negatif (*background*). Untuk data positif pada citra, posisi objek ditandai dengan

bounding box yang melingkupi objek. Dalam melakukan *generate* sampel data *training* positif dari database, semua citra objek *dicrop* disekitar *bounding box* masing-masing, disejajarkan dan ukurannya dinormalisasi. Sebaliknya pada data negatif, sampel diambil secara random dari citra *background* dengan ukuran yang sama dengan sampel positif. Contoh dataset positif dan negatif dapat dilihat pada gambar 2.11 dan 2.12.

2.3.2.2. Testing classifier

Setelah dataset *training* didapatkan, selanjutnya dilakukan pembangunan *classifier* seperti pada Subbab 2.3.1. Setelah *classifier* didapatkan, selanjutnya dilakukan evaluasi performa *classifier* terhadap semua dataset *training*. Hasil evaluasi performa *classifier* merupakan citra original yang telah diberi label. Selama proses evaluasi, *classifier* dites dalam beberapa skala untuk setiap lokasi citra. Hasil evaluasi ini akan menghasilkan deteksi objek potensial \tilde{x}_i yang memiliki asosiasi dengan masing-masing *confidence* level. Ilustrasi skala yang dilakukan dalam *testing* ditampilkan pada Gambar 2.14.



Gambar 2.14 Ilustrasi skala citra dalam *testing* (Villamizar et al., 2012).

Pada gambar 2.14., *classifier* (kotak merah) dievaluasi pada setiap lokasi dan terhadap beberapa resolusi HOG. Untuk meminimalisir *cost* komputasi, HOG *pyramid* dihitung menggunakan citra integral.

Kandidat objek dengan *confidence* level terbesar diambil dan diberi label sebagai sampel *true* positif atau sampel *false* positif (negatif)

berdasarkan derajat *overlapping* terhadap posisi *ground truth* objek. Persamaan untuk menghitung derajat *overlapping* ditampilkan pada persamaan 2.14.

$$r = \frac{B(\tilde{x}_i) \cap B_{true}}{B(\tilde{x}_i) \cup B_{true}} \quad (2.14)$$

Pada persamaan 2.14, r merupakan rasio area, sedangkan B_{true} merupakan *ground truth bounding box*. Sebuah sampel objek \tilde{x}_i dalam *bounding box* $B(\tilde{x}_i)$ dianggap sebagai *true* positif jika nilai r lebih besar dari r_{max} . Sebaliknya jika r lebih kecil dari r_{max} maka dianggap sebagai sampel negatif. Nilai r_{max} yang biasa digunakan adalah 0.5, kecuali untuk eksperimen khusus yang membutuhkan nilai r_{max} berbeda.

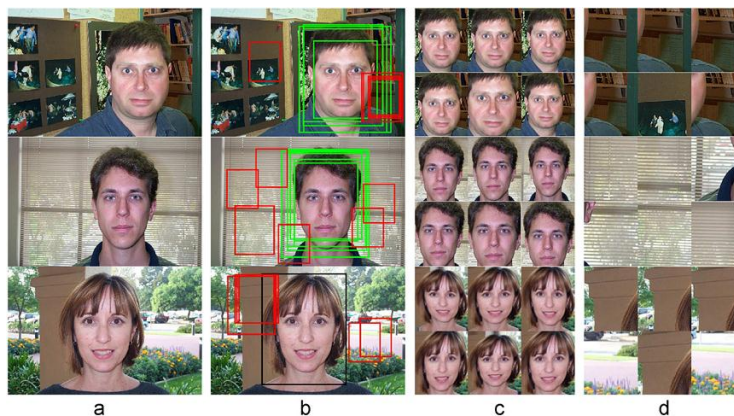
2.3.2.3. Menambahkan sampel dataset *training* baru

Setelah melakukan *testing classifier* terhadap seluruh dataset, didapatkan satu set sampel yang dapat diklasifikasikan sebagai data positif maupun negatif. Sebagai tambahan juga terdapat subset sampel objek yang tidak terdeteksi yang berasal dari *ground truth* yang tidak dideteksi sebagai data positif oleh *classifier*. Selanjutnya untuk meningkatkan *recognition rate* dari *classifier*, ditambahkan sampel objek yang didapatkan ke dalam dataset citra *training* original. Kriteria sampel yang ditambahkan dalam dataset *training* terbagi dalam tiga jenis berikut :

- Sampel true positif ditambahkan ke dalam dataset *training* positif beserta ukuran skala dan *offset* di mana sampel tersebut terdeteksi. Hal ini untuk memastikan sampel yang terdeteksi tidak hilang pada proses *training* berikutnya.
- Sampel false positif berupa *background* yang dideteksi sebagai objek, ditambahkan ke dalam dataset *training* negatif.
- Sampel tidak terdeteksi ditambahkan ke dalam dataset *training* positif. Untuk meningkatkan fokus *classifier* terhadap sampel ini,

sampel akan digandakan dalam beberapa jenis tranformasi yang berbeda seperti perubahan kemiringan dan skala.

Dalam setiap proses *bootstrapping*, ukuran *training* set akan semakin bertambah. Oleh karena itu, untuk mengontrol jumlah dataset maka untuk setiap sampel dibatasi pengambilan kandidat object sebanyak 6 sampel di mana sampel true positif sampel akan langsung ditambahkan ke dalam dataset *training* positif, sedangkan untuk setiap sampel tidak terdeteksi dan sampel false positif akan dilakukan generate 6 data *training* baru. Misalkan terdapat dataset *training* dengan jumlah sekitar 50 data, maka akan berkembang menjadi sekitar 600 data dalam dua proses *bootstrapping*. Namun tentu saja untuk dataset yang lebih besar, maka parameter ini akan berubah untuk menjaga dataset *training* agar tetap mudah digunakan. Ilustrasi pengambilan dataset baru ditampilkan dalam Gambar 2.15. di mana bagian a adalah citra original, bagian b adalah hasil deteksi oleh *classifier*, bagian c positif sampel sedangkan bagian d adalah negatif sampel.

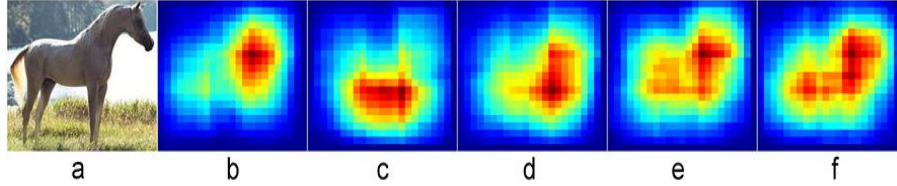


Gambar 2.15 Ilustrasi pengambilan dataset baru (Villamizar et al., 2012).

2.3.2.4. Melakukan update *classifier*

Selanjutnya ketika dilakukan pencarian *weak classifier* dan *strong classifier* terhadap dataset *training* baru, maka akan diperoleh *classifier* yang lebih berdedikasi terhadap permasalahan sampel (lebih sesuai terhadap

sampel). Kemudian konsekuensi dari perubahan *classifier* adalah lebih menekankan terhadap ciri-ciri khusus dari objek sehingga akan mengurangi *global classification error*. Ilustrasi kelebihan *boostapping* ditampilkan dalam Gambar 2.16.



Gambar 2.16 Ilustrasi distribusi *random ferns* (Villamizar et al., 2012).

Gambar 2.16 menggambarkan layout spasial distribusi *random ferns* dalam proses *boostapping* terhadap dataset kuda. Gambar 2.16, bagian a merupakan citra original, sedangkan Gambar 2.16 bagian b sampai f merupakan layout spasial distribusi *random ferns* dalam proses *boostapping* dari iterasi ke-0 sampai ke-4. Kemudian digunakan beberapa warna berbeda untuk menggambarkan persebaran distribusi *random ferns*. Semakin mendekati warna merah, maka semakin tinggi konsentrasi random fern dan semakin mendekati warna biru, maka semakin rendah konsentrasi *random ferns*. Pada iterasi ke-0, persebaran fern lebih condong hanya pada sekitar leher kuda, kemudian dengan semakin bertambahnya iterasi, fitur diskriminatif semakin berkembang terhadap bagian tubuh kuda.

2.4. Tanimoto Distance

Tanimoto distance merupakan metode pengujian terhadap klasifikasi *multilabel*. *Tanimoto distance* mampu melakukan evaluasi terhadap jumlah total kategori yang berhasil diprediksi secara benar oleh algoritma klasifikasi (Oliveira et al., 2008). Persamaan yang digunakan dalam *tanimoto distance* ($tanimoto_j$) ditampilkan dalam persamaan 2.8.

$$tanimoto_j = \frac{|P_j| + |C_j| - 2|P_j \cap C_j|}{|P_j| + |C_j| - |P_j \cap C_j|} \quad (2.15)$$

Tabel 2.1 Contoh hasil prediksi

Nama Citra	Motif	
	Hasil Prediksi (P_j)	Sebenarnya (C_j)
Batik 1	parang-semen	parang-semen-buket
Batik 2	parang-megamendung	parang-megamendung

Pada persamaan 2.8, $P_j \cap C_j$ merupakan irisan antar kategori yang diprediksi (P_j) dan kategori sebenarnya (C_j). Kategori yang diprediksi adalah hasil dari klasifikasi yang dilakukan algoritma, sedangkan kategori sebenarnya merupakan kategori yang sebenarnya terdapat dalam citra. Semakin banyak banyak jumlah kategori P_j yang sama dengan kategori C_j , maka nilai $tanimoto_j$ akan semakin mendekati 0. Contoh kasus untuk tanimoto distance ditampilkan dalam Tabel 2.1.

Pada Tabel 2.1 untuk citra Batik 1 memiliki nilai $P_j=2$ (parang, semen) dan nilai $C_j=3$ (parang, semen, buket) sedangkan nilai irisan $P_j \cap C_j=2$ (parang, semen). Berdasarkan nilai yang diketahui, maka $tanimoto_j$ dari citra Batik 1 adalah sebagai berikut :

$$tanimoto_j \text{ citra Batik 1} = \frac{2 + 3 - 2(2)}{2 + 3 - 2} = 0.33$$

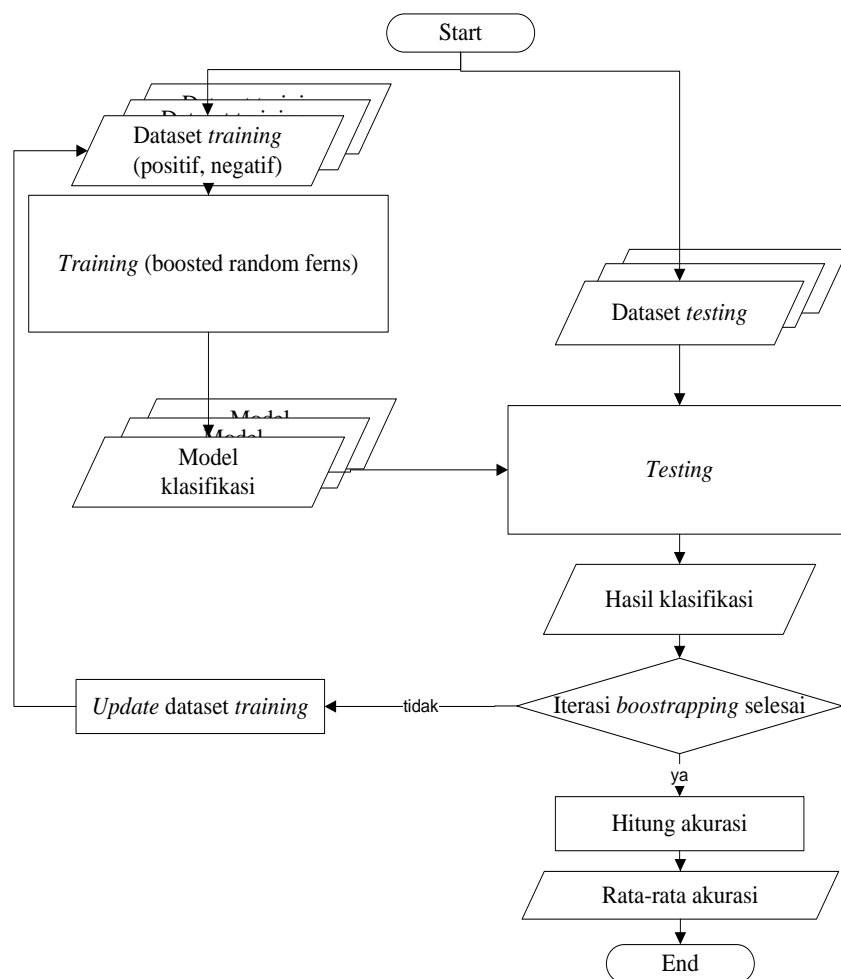
Sedangkan untuk citra Batik 2 memiliki nilai $P_j=2$ (parang, megamendung) dan nilai $C_j=2$ (parang, megamendung) sedangkan nilai irisan $P_j \cap C_j=2$ (parang, megamendung). Berdasarkan nilai yang diketahui, maka $tanimoto_j$ dari citra Batik 2 adalah sebagai berikut :

$$tanimoto_j \text{ citra batik 2} = \frac{2 + 2 - 2(2)}{2 + 2 - 2} = 0.$$

BAB 3

METODOLOGI PENELITIAN

Pada bab ini dijelaskan tentang metodologi penelitian meliputi desain sistem yang dibangun, skenario pengujian serta dokumentasi dan penjadwalan. Sistem klasifikasi *multilabel boosted random ferns* yang akan dibangun terdiri dari dua proses utama yakni proses *training* dan proses *testing*. Proses *training* dilakukan untuk menyusun model klasifikasi yang akan digunakan. Sedangkan proses *testing* dilakukan untuk mengetahui seberapa akurat model klasifikasi yang disusun. Desain sistem klasifikasi *multilabel boosted random ferns* yang akan dibangun ditampilkan dalam *flowchart* pada Gambar 3.1.



Gambar 3.1 Flowchart desain sistem klasifikasi *multilabel boosted random ferns*.

Pada proses *training*, *input* yang digunakan berupa dataset *training* yang terdiri dari dataset motif batik parang, kawung, megamendung, nitik, semen, lunglungan dan dataset *background*. Dataset-dataset tersebut selanjutnya disusun menjadi dataset positif dan negatif ke dalam dataset *training* motif parang, kawung, megamendung, nitik, semen dan lunglungan. Dataset positif berupa citra objek dengan motif batik yang sesuai dengan kelas motifnya. Sedangkan dataset negatif berupa citra *background* maupun citra motif batik yang tidak sesuai dengan kelas motifnya.

Selanjutnya dilakukan proses *training* terhadap seluruh dataset *training*. Proses *training* dilakukan untuk membentuk model klasifikasi dari masing-masing motif batik (parang, kawung, megamendung, nitik, semen dan lunglungan). Model-model klasifikasi ini selanjutnya digunakan dalam proses *testing* sebagai subproses untuk mengidentifikasi motif pada data *testing*. Penjelasan proses *training* akan dijelaskan secara detail dalam subbab 3.1.

Sedangkan dalam proses *testing*, *input* yang digunakan adalah dataset *testing* yang berupa citra batik yang terdiri dari satu motif batik maupun kombinasi beberapa motif batik. Jenis motif batik yang digunakan sebagai *input* dalam proses *testing* terbatas pada 6 motif batik, yakni motif batik parang, kawung, megamendung, nitik, semen dan lunglungan. Selanjutnya model-model klasifikasi motif akan mengidentifikasi motif-motif yang sesuai dengan model klasifikasi motifnya. Misalkan model klasifikasi motif parang akan melakukan identifikasi motif parang, sedangkan model klasifikasi motif kawung akan melakukan identifikasi motif kawung. Begitu juga dalam model klasifikasi motif lainnya, masing-masing mengidentifikasi motif sesuai model klasifikasi motifnya.

Hasil identifikasi model-model klasifikasi motif tersebut berupa *bounding box* yang menandai lokasi bentuk motif yang ditemukan. *Bounding box* hasil klasifikasi selanjutnya dicocokkan dengan *bounding box ground truth* pada citra untuk melakukan pelabelan *true* positif, *false* positif dan *unidentified*. Pelabelan ini dilakukan untuk mengupdate dataset *training* menggunakan subproses *bootstrapping*. Subproses *bootstrapping* akan melakukan iterasi sebanyak 4 kali, di

mana pada setiap iterasi dilakukan *update* dataset *training* berdasarkan label *true* positif, *false* positif dan *unidentified*.

Setelah iterasi *bootstrapping* selesai, dilakukan pelabelan motif berdasarkan hasil identifikasi motif dari model-model klasifikasi motif batik. Pelabelan ini berupa teks jenis motif. Misalkan pada suatu citra hanya diidentifikasi memiliki motif kawung oleh model klasifikasi kawung, maka akan dilabeli kawung. Sedangkan jika suatu citra diidentifikasi memiliki motif parang oleh model klasifikasi parang dan diidentifikasi memiliki motif kawung oleh model klasifikasi motif kawung, maka akan dilabeli parang-kawung. Begitu juga jika diidentifikasi memiliki motif-motif yang lain akan dilabeli sesuai jenis motifnya.

Label-label motif hasil identifikasi kemudian akan dihitung akurasi terhadap *ground truth* motif sebenarnya. Hasil rata-rata akurasi dari keseluruhan dataset *testing* menjadi hasil *output* dari proses *testing*. Penjelasan proses *testing* akan dijelaskan secara detail dalam subbab 3.2.

3.1. Proses Training

Proses *training boosted random ferns* memiliki *input* 340 data *training* yang terdiri dari 40 citra motif parang, 40 citra motif kawung, 30 citra motif lunglungan, 30 citra motif megamendung, 50 citra motif nitik, 30 citra motif semen, dan 120 citra background. Perbedaan jumlah citra dalam dataset motif dikarenakan perbedaan jumlah variasi setiap motif, pada motif parang terdapat 4 variasi, motif kawung 4 variasi, motif lunglungan 3 variasi, motif megamendung 3 variasi, motif nitik 5 variasi dan motif semen 3 variasi. Seluruh citra dalam dataset *training* memiliki ukuran 50x50 piksel.

Selanjutnya dataset *training* dipilah menjadi dataset positif dan negatif dalam masing-masing motif parang, kawung, megamendung, nitik, semen dan lunglungan. Dataset positif berupa citra objek dengan motif batik yang sesuai dengan kelas motifnya. Sedangkan dataset negatif berupa citra *background*

maupun citra motif batik yang tidak sesuai dengan kelas motifnya. Pemilahan dataset akan dijelaskan lebih lanjut dalam subbab 3.1.1.

Kemudian dilakukan penyusunan model klasifikasi motif terhadap dataset *training* yang telah dipilah. Penyusunan model klasifikasi motif memiliki tiga tahap, yakni pembangkitan *weak classifier*, *update* bobot *weak classifier* dan pembangkitan *strong classifier*. Pembangkitan *weak classifier* digunakan untuk menyusun model klasifikasi sederhana namun tidak begitu akurat. *Weak classifier* akan melakukan *update* bobot terhadap data *training* jika terdapat kesalahan (*error*) dalam identifikasi data *training*. Data yang salah diidentifikasi, bobotnya ditambah, sedangkan data yang sudah teridentifikasi dengan benar, bobotnya dikurang. Perubahan bobot digunakan untuk membuat data yang salah diidentifikasi mendapatkan perhatian lebih.

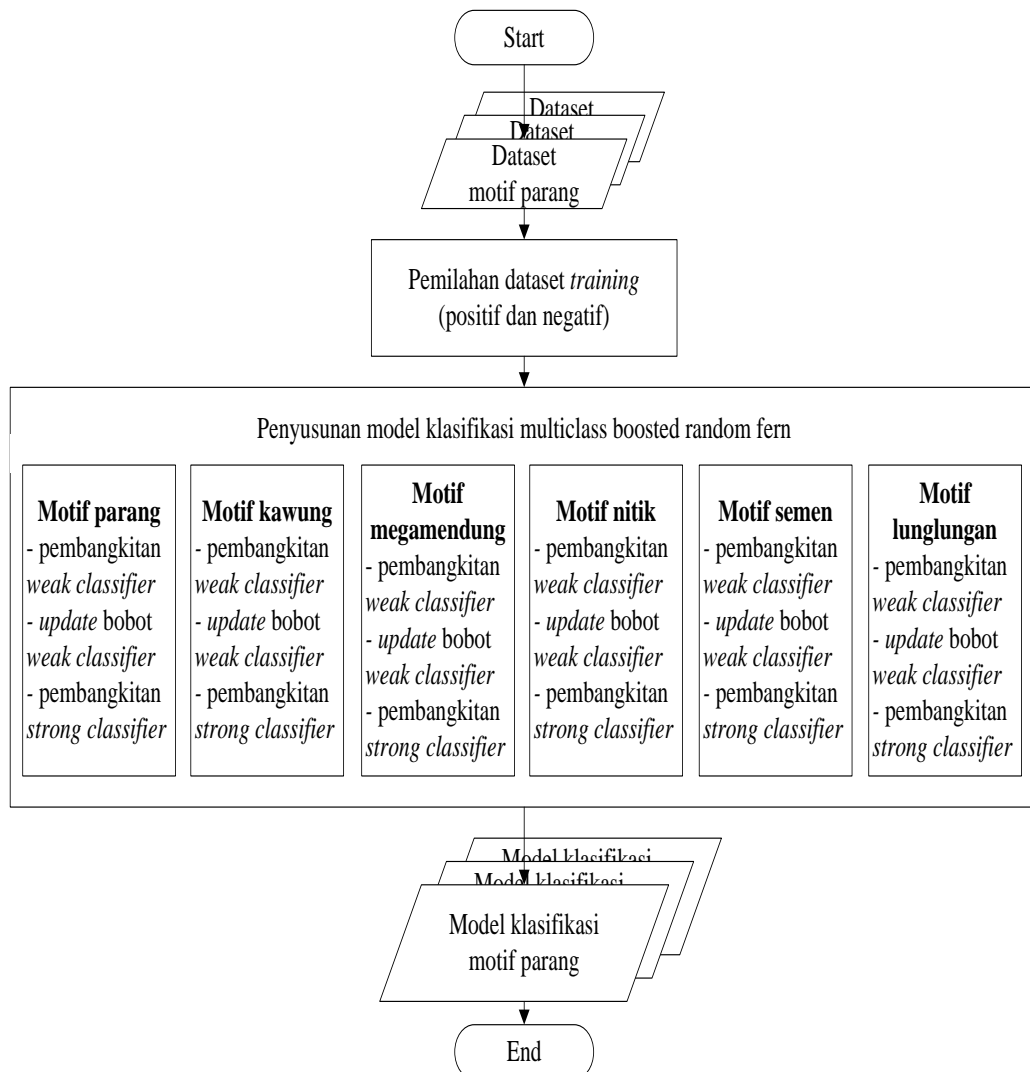
Weak classifier tidak mampu melakukan klasifikasi secara akurat jika sendirian. *Weak classifier* harus dikombinasikan dengan *weak classifier-weak classifier* lainnya untuk membangkitkan *strong classifier* yang mampu melakukan klasifikasi secara akurat. *Strong classifier* selanjutnya digunakan sebagai model klasifikasi motif. Pembangkitan *weak classifier*, *update* bobot *weak classifier* dan pembangkitan *strong classifier* akan dijelaskan dalam subbab 3.1.2, 3.1.3 dan 3.1.4.

Hasil *output* dari proses *training* adalah 6 model klasifikasi, yakni model klasifikasi motif batik parang, kawung, megamendung, nitik, semen dan lunglungan. Model-model klasifikasi motif akan digunakan untuk melakukan identifikasi motif-motif pada citra *testing* dalam proses *testing*. Alur proses *training* ditampilkan dalam *flowchart* dalam Gambar 3.2.

3.1.1. Pemilahan dataset

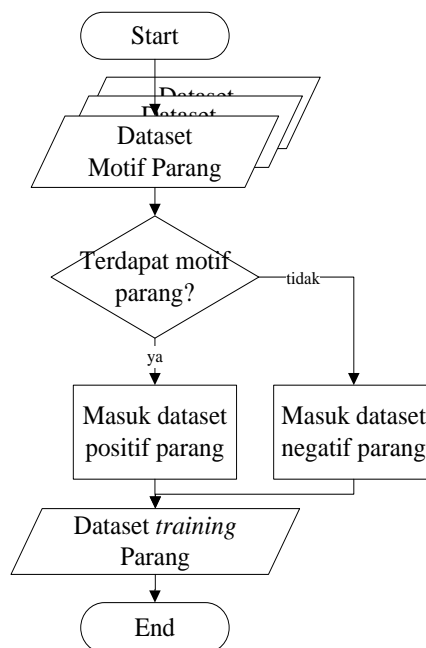
Pemilahan dataset dilakukan untuk menyediakan dataset *training* untuk model-model klasifikasi motif. Setiap model klasifikasi motif memiliki susunan dataset *training* positif dan negatif yang berbeda. Perbedaan susunan dataset *training* positif dan negatif terjadi karena

perbedaan identifikasi yang dilakukan model-model klasifikasi motif. Misalkan model klasifikasi motif parang yang melakukan identifikasi motif parang. Dataset positif model klasifikasi motif parang hanya berisi dataset motif parang, sedangkan dataset negatif model klasifikasi motif parang berisi dataset motif kawung, megamendung, nitik, semen dan lunglungan maupun dataset *background*. Begitu juga dengan dataset model klasifikasi motif yang lain, masing masing memiliki dataset positif yang berisi dataset motifnya sendiri dan dataset negatif yang berisi dataset *background* dan dataset motif selain motifnya sendiri.



Gambar 3.2. Flowchart proses training.

Dataset *training* original (sebelum mengalami *update* dataset oleh subproses *bootstrapping*) memiliki jumlah data 340 citra *training* yang terdiri dari 6 dataset motif batik (parang 40 citra, kawung 40 citra, lunglungan 30 citra, megamendung 30 citra, nitik 50 citra dan semen 30 citra) dan 1 dataset *backgorund* (120 citra). Seluruh citra dalam dataset *training* memiliki ukuran 50x50 piksel. Dataset motif batik terdiri dari data citra yang termasuk salah satu dari motif parang, kawung, megamendung, nitik, semen atau lunglungan, sedangkan dataset *background* berisi data citra yang tidak termasuk citra motif apapun. Perbedaan jumlah citra dalam dataset motif dikarenakan perbedaan jumlah variasi setiap motif, pada motif parang terdapat 4 variasi, motif kawung 4 variasi, motif lunglungan 3 variasi, motif megamendung 3 variasi, motif nitik 5 variasi dan motif semen 3 variasi. Variasi ini akan ditampilkan dalam Tabel 3.1. Ukuran semua citra dalam masing-masing dataset adalah 50x50 piksel. Alur proses pemilahan dataset *training* motif parang diilustrasikan pada *flowchart* pada Gambar 3.3.

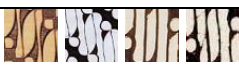
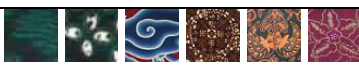







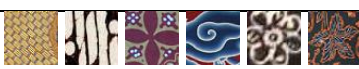

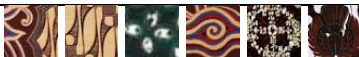


Gambar 3.3 *Flowchart* pemilahan dataset parang

Dataset *training* dipilah secara manual ke dalam 6 susunan dataset *training* motif batik (parang, kawung, megamendung, nitik, semen dan lunglungan). Setiap susunan dataset motif batik terdiri dari dataset positif dan dataset negatif. Dataset positif adalah data citra yang dianggap sebagai objek sedangkan dataset negatif adalah data citra yang dianggap sebagai *background*. Pada susunan dataset motif parang, dataset positif berisi dataset motif parang, sedangkan dataset negatif berisi dataset *background* dan dataset motif selain parang. Begitu juga susunan dataset motif yang lain, dataset positif berisi dataset motif yang sesuai dengan motifnya sedangkan dataset negatif berisi dataset *background* dan dataset yang tidak sesuai motifnya.

Setiap susunan dataset *training* motif batik digunakan sebagai dataset *training* model klasifikasi motif batik sesuai motifnya. Seperti susunan dataset *training* motif parang digunakan sebagai dataset *training* model klasifikasi motif parang, dan susunan dataset *training* motif kawung digunakan sebagai dataset *training* model klasifikasi motif kawung. Untuk proses pemilahan dataset *training* motif yang lain, menggunakan *flowchart* pada Gambar 3.3 dengan menyesuaikan motif masing-masing.

Tabel 3.1 Tabel Contoh Hasil Pemilahan Dataset

No	Motif	Positif	Negatif
1	Parang		
2	Kawung		
3	Megamendung		
4	Nitik		
5	Semen		
6	Lunglungan		

3.1.2. Pembangkitan *weak classifier*

Pembangkitan *weak classifiers* merupakan dasar dari penyusunan model klasifikasi motif batik. Pembangkitan *weak classifier* digunakan untuk menyusun model klasifikasi sederhana namun tidak begitu akurat. *Weak classifier* menggunakan perhitungan klasifikasi *random ferns* terhadap ruang bin *histogram of oriented gradient* (HOG). Kumpulan *weak classifier* yang paling optimal akan dikombinasikan menjadi *strong classifier*.

Pembangkitan *weak classifier* menjadi bagian dalam langkah-langkah penyusunan model klasifikasi motif batik. Sebagai contoh, digunakan penyusunan model klasifikasi motif parang. Berikut langkah-langkah penyusunan model klasifikasi motif parang:

1. Setelah citra dataset *training* motif parang diinputkan, dilakukan penentuan bobot awal dari tiap citra dalam dataset motif parang, jumlah *weak classifier*, jumlah *subset random ferns* dan posisi dari 7 pasang bin yang dibandingkan dalam *subset random ferns*. Bobot awal dari tiap citra adalah $1/\text{jumlah citra dalam dataset motif parang}$ sehingga bobot awal adalah $1/700=0,0014$. Sedangkan jumlah *weak classifier* sebanyak 100 dan jumlah *subset random ferns* 5 *subset*.
2. Selanjutnya dilakukan pemilihan 5 posisi piksel dari citra dataset *training* motif parang secara random untuk sebagai pusat 5 *subset random ferns* (*subset F*) beserta 7 pasang bin HOG yang dipilih secara random dari bin 0 sampai bin 35 dengan pasangan yang tidak boleh berulang. Posisi piksel dan 7 pasang bin HOG yang dipilih disimpan dan digunakan secara konsisten pada keseluruhan citra dataset *training* motif pada *weak classifier* saat ini. Namun dalam *weak classifier* selanjutnya posisi piksel dan 7 pasang bin HOG akan dipilih lagi secara random. Terdapat batasan dari posisi piksel yang dipilih, yakni harus memiliki jarak lebih dari 8 piksel

dari setiap tepi citra. Batasan tersebut karena ukuran minimum dari *block* HOG adalah 16x16 piksel.

3. Setelah 5 *subset F* memiliki pusat, dibentuk 5 *region* citra berukuran 16x16 piksel berdasarkan pusat piksel masing-masing.
4. Kemudian dilakukan ekstraksi fitur HOG setiap *subset F*. Ekstraksi fitur HOG dijelaskan lebih lanjut dalam subbab 3.2.1. Hasil ekstraksi fitur HOG ini merupakan vektor bin histogram 1 dimensi berukuran 1x36 bin.
5. Berikutnya dilakukan perbandingan 7 pasang bin HOG. Jika lebih besar dari pasangannya akan diberi nilai 1 sedangkan jika sama dengan atau lebih kecil dari pasangannya maka akan diberi nilai 0. Selanjutnya seluruh hasil perbandingan dijadikan satu vektor dan menjadi nilai fitur *random ferns*. Nilai fitur ini berupa vektor biner dengan panjang 1x7 biner yang mewakili satu citra.
6. Berdasarkan fitur-fitur *random ferns* yang telah didapatkan, selanjutnya dihitung *weak classifier* dari masing-masing fitur *random fern* pada *subset F*. *Weak classifier* dihitung berdasarkan pembagian probabilitas kemunculan fitur *random ferns* terhadap dataset positif parang oleh probabilitas kemunculan fitur *random ferns terhadap* dataset negatif parang. Untuk menghindari probabilitas kemunculan yang bernilai 0, maka ditambahkan 1 kemunculan pada kedua probabilitas. Hasil pembagian selanjutnya dikalikan dengan $\frac{1}{2} \log$. Hasil perhitungan tersebut menjadi nilai *weak classifier* dari *subset F*.
7. Selanjutnya masing-masing *weak classifier* dari 5 *subset F* dihitung performa klasifikasinya menggunakan *bhattacharya distance*. *Bhattacharya distance* dilakukan dengan mengkalikan probabilitas kemunculan masing-masing fitur *random ferns* terhadap dataset positif parang dengan probabilitas kemunculan masing-masing fitur *random ferns terhadap* dataset negatif parang. Selanjutnya hasil

perkalian tersebut diakar dan dijumlahkan dengan hasil perhitungan probabilitas fitur *random ferns* yang lain.

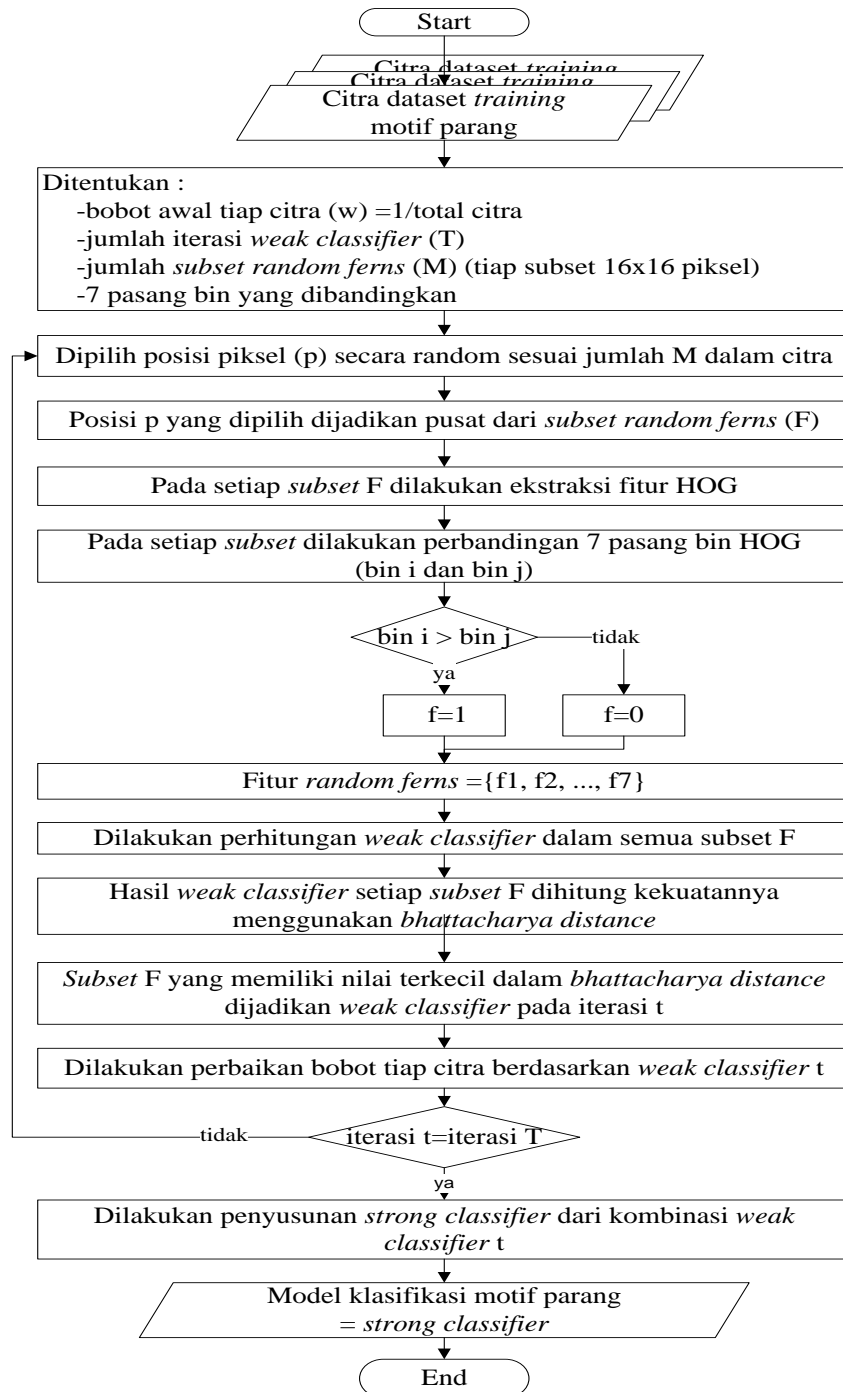
8. *Weak classifier* dari fitur *random ferns subset F* yang memiliki nilai *bhattacharya distance* terkecil akan dipilih sebagai *weak classifier t* (*weak classifier* saat ini). Posisi piksel pusat dari *subset F* yang dipilih disimpan untuk proses pemetaan *subset* dan perhitungan *strong classifier*.
9. Selanjutnya dilakukan perubahan bobot tiap citra *training* berdasarkan hasil klasifikasi *weak classifier t*. Penjelasan perubahan bobot akan dijelaskan dalam subbab 3.1.3.
10. Dilakukan iterasi sampai sebanyak jumlah *weak classifier* yang digunakan (100). Iterasi dilakukan terhadap langkah nomor 2 sampai 9.
11. Setelah iterasi selesai, *strong classifier* disusun berdasarkan kombinasi *weak classifier t*. Penyusunan *strong classifier* dilakukan dengan cara melakukan voting terhadap hasil klasifikasi *weak classifier-weak classifier t*. Penjelasan lebih lanjut dijelaskan dalam subbab 3.1.4.
12. Hasil akhir yang didapatkan adalah model klasifikasi motif parang yang berupa *strong classifier*.

Langkah-langkah penyusunan model klasifikasi motif parang berlaku untuk penyusunan model klasifikasi motif lainnya dengan dataset *training* masing-masing motif. Alur dari langkah-langkah model klasifikasi motif parang ditampilkan dalam *flowchart* pada Gambar 3.4.

3.1.2.1. Histogram of oriented gradient

Vektor HOG merupakan vektor bin orientasi yang berisi nilai gradien dari citra. Langkah pertama dari perhitungan HOG yaitu dengan mengkalikan filter I_x $[-1 \ 0 \ 1]$ dan I_y $\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$ terhadap subset F citra. Hasil dari perkalian ini merupakan nilai gradien dari citra. Kemudian dilakukan

pencarian nilai orientasi citra θ yang diperoleh dengan melakukan operasi $90 + (\arctan(\frac{I_x}{I_y}))$.



Gambar 3.4 Flowchart penyusunan model klasifikasi motif parang.

Setelah nilai gradien dan orientasi diperoleh, setiap piksel dalam subset F akan memiliki nilai gradien dan nilai orientasi. Selanjutnya citra dibagi dalam *region cell* yang berisi 8x8 piksel yang tidak saling overlapping. Nilai gradien dalam *cell* kemudian dijumlahkan ke dalam bin orientasi yang sesuai dengan nilai orientasi pikselnya.

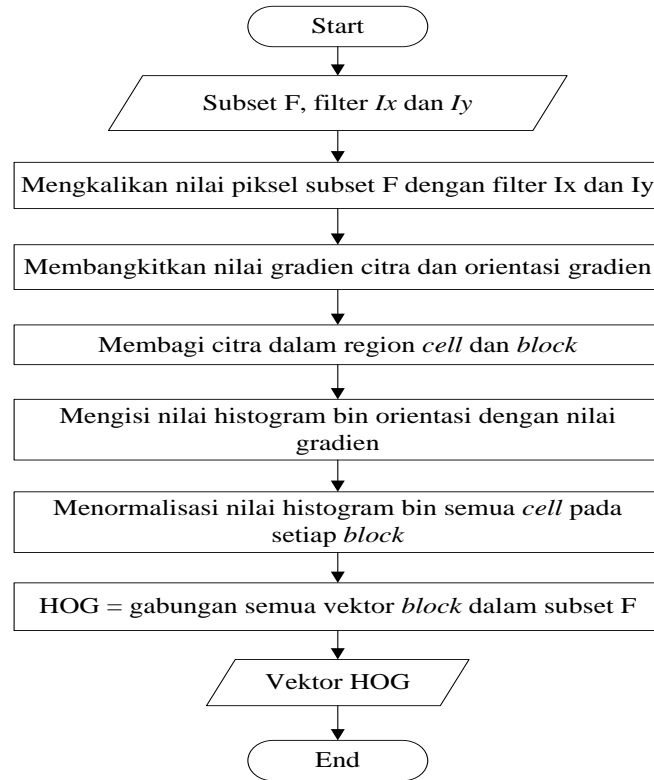
Terdapat 9 bin orientasi dari bin 0 sampai bin 8. Setiap bin orientasi memiliki *range* sebesar $180/9 = 20$. Pada setiap range terdapat bin center yang berada ditengah-tengah setiap bin. Pembagian bin beserta bin center masing-masing ditampilkan dalam Gambar 3.5.



Gambar 3.5 Bin orientasi HOG.

Misalkan nilai gradien 80 memiliki nilai orientasi 65° , maka gradien tersebut akan dijumlahkan ke dalam bin 2 sebesar 20 dan bin 3 sebesar 60. Pembagian nilai tersebut berdasarkan jarak kedekatan dengan bin center. Nilai 65° memiliki jarak sebesar 15° dari center bin 2 dan 5° dari center bin 3. Untuk bin 2 diberlakukan perhitungan $80 * \frac{20-15}{20} = 20$, sedangkan untuk bin 3 diberlakukan perhitungan $80 * \frac{20-5}{20} = 60$, dimana angka 20 diperoleh dari *range* bin.

Selanjutnya subset F dibagi dalam *region block* yang berukuran 16x16 piksel dan saling *overlapping* pada setiap 8x8 piksel (satu *cell*). Setiap *block* berisi 2x2 *cell*. Nilai bin semua *cell* dalam setiap block kemudian dinormalisasi. Hasil normalisasi semua block dalam subset F kemudian digabung dan menjadi vektor HOG. Ilustrasi dari alur HOG ditampilkan dalam Gambar 3.6.



Gambar 3.6 Flowchart HOG.

3.1.3. Update bobot weak classifier

Weak classifier akan melakukan *update* bobot terhadap data *training* jika terdapat kesalahan (*error*) dalam identifikasi data *training*. Data yang salah diidentifikasi, bobotnya ditambah, sedangkan data yang sudah teridentifikasi dengan benar, bobotnya dikurang. Perubahan bobot digunakan untuk membuat data yang salah diidentifikasi mendapatkan perhatian lebih. Perubahan bobot (W) dilakukan untuk memperbaiki kesalahan *weak classifier* sebelumnya. Perubahan W dihitung dengan persamaan 3.1.

$$W^{t+1}(x_i) = \frac{W^t(x_i) \exp(y_i h_C^t(x_i))}{\sum_{i=1}^N W^t(x_i) \exp(y_i h_C^t(x_i))} \quad (3.1)$$

Di mana W^{t+1} merupakan perubahan bobot sampel x_i dari dataset

training dan y_i adalah indikasi sampel adalah dataset positif (bernilai 1) atau negatif (bernilai -1). Sedangkan h_c^t adalah *weak classifier* pada motif tertentu. Hasil dari perubahan bobot ini akan menentukan prioritas klasifikasi terhadap dataset *training*.

3.1.4. Pembangkitan *strong classifier*

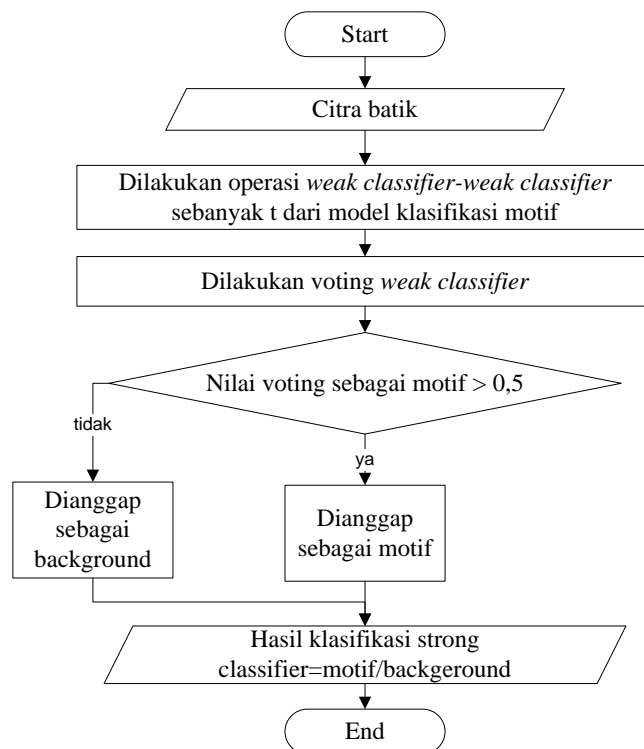
Strong classifier didapatkan dengan cara melakukan kombinasi *weak classifier-weak classifier* yang terpilih dari setiap iterasi t . *Weak classifier* tersebut dihitung terhadap fitur *random ferns* yang terdapat dalam *subset random ferns*. Contoh distribusi *subset random ferns* yang menjadi *weak classifier* pada iterasi t ditampilkan dalam Gambar 3.7.



Gambar 3.7 Distribusi *subset random ferns* (Villamizar et.al, 2012).

Gambar 3.7 menampilkan distribusi posisi *subset random ferns* yang terpilih menjadi *weak classifier* t . Terdapat variasi intensitas *subset random ferns* dalam Gambar 3.7, semakin mendekati warna merah, semakin banyak *subset random ferns* pada posisi tersebut. *Weak classifier-weak classifier* dalam setiap *subset random ferns* yang terpilih dalam setiap iterasi t . Kemudian karena terdapat 6 jenis motif yang digunakan, maka akan terdapat 6 *strong classifier* yang dihasilkan untuk membedakan dataset positif dan negatif masing-masing motif. *Strong classifier* ini selanjutnya digunakan sebagai model klasifikasi motif batik.

Kemudian dalam melakukan klasifikasi, *strong classifier* menggunakan hasil voting dari hasil klasifikasi *weak classifier-weak classifier* t. Hasil klasifikasi yang memiliki nilai voting terbesar akan digunakan sebagai hasil klasifikasi dari *strong classifier*. Alur dari proses klasifikasi strong classifier ditampilkan dalam Gambar 3.8.



Gambar 3.8 Flowchart klasifikasi dengan *strong classifier*

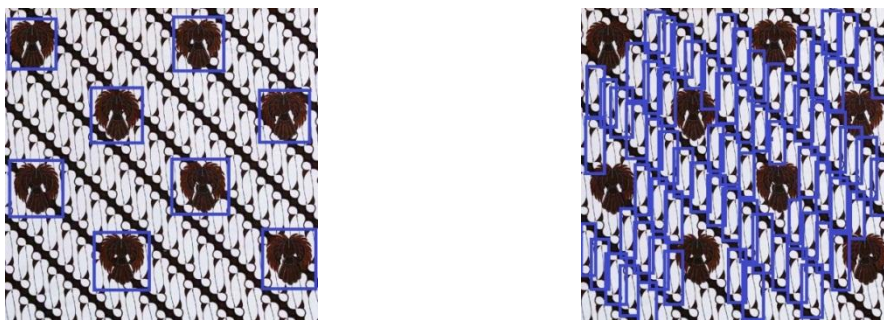
3.2. Proses Testing

Dalam proses *testing*, *input* yang digunakan adalah dataset testing motif batik, di mana setiap citra memiliki satu atau beberapa motif batik. Digunakan 64 citra batik yang terbagi menjadi 42 citra motif satuan dan 22 citra motif kombinasi. Citra motif satuan terdiri dari 6 citra parang, 9 citra kawung, 6 citra lunglungan, 9 citra megamendung, 6 citra semen dan 6 citra nitik. Citra motif kombinasi terdiri dari 2 citra kombinasi parang-kawung-lunglungan, 4 citra kombinasi parang-megamendung, 7 citra kombinasi parang-nitik, 3 citra kombinasi parang-semen, 3 citra kombinasi kawung-nitik dan 3 citra kombinasi

semen lunglungan. Dalam proses ini, dataset akan dideteksi menggunakan model klasifikasi masing-masing motif dataset.

Untuk mengetahui kebenaran dari hasil deteksi, setiap dataset *testing* diberi *ground truth* motif batik berupa *bounding box* yang melingkupi motif-motif pada setiap citra. *Ground truth* untuk setiap motif dibuat terpisah karena proses klasifikasi dari model-model klasifikasi motif batik juga dilakukan secara terpisah. Contoh *ground truth* secara terpisah seperti pada Gambar 3.9, di mana pada gambar sebelah kanan adalah *ground truth* dari motif semen (ditandai dengan kotak biru) dan gambar sebelah kiri adalah *ground truth* dari motif parang.

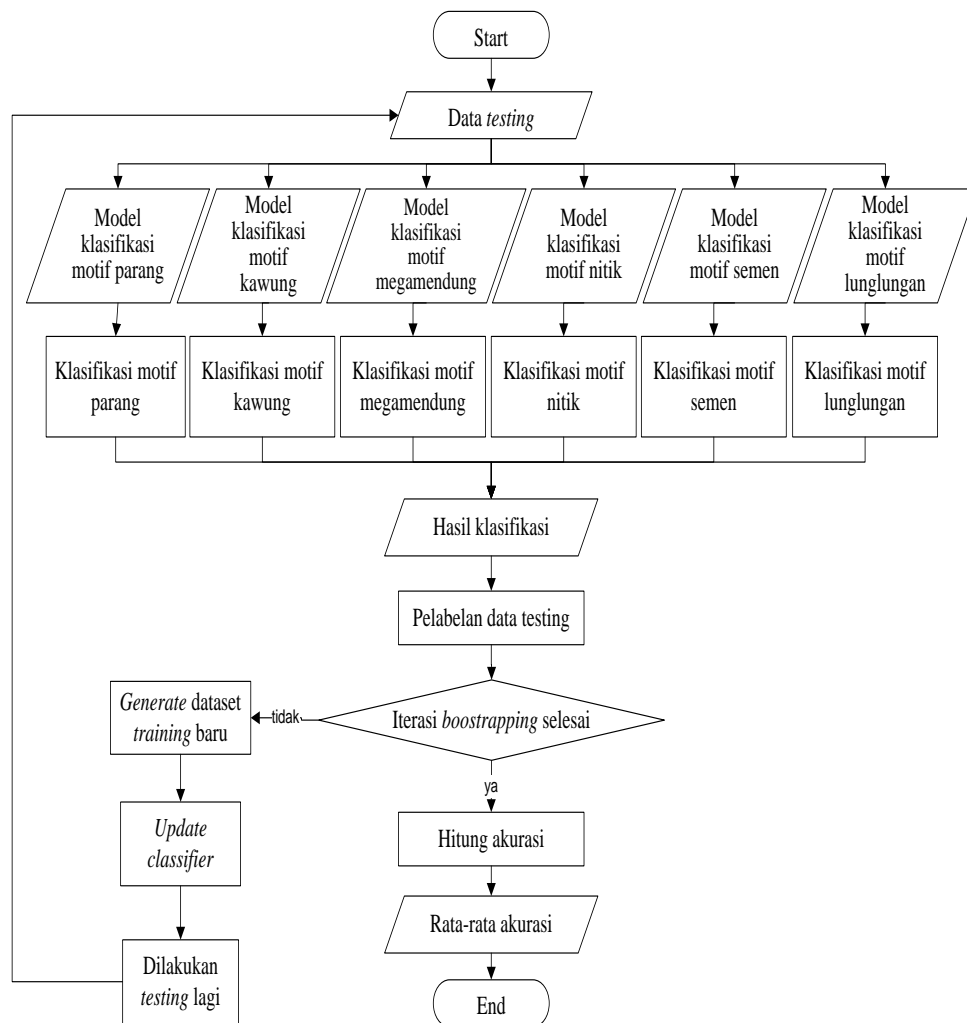
Selanjutnya masing-masing model klasifikasi motif batik akan melakukan deteksi terhadap masing-masing motif. Proses klasifikasi menggunakan model klasifikasi motif batik akan dijelaskan lebih lanjut dalam subbab 3.2.1. Hasil deteksi dari masing-masing model klasifikasi motif akan diberi label sesuai perbandingan hasil deteksi dengan *ground truth*. Pelabelan ini akan dijelaskan lebih lanjut pada subbab 3.2.2. Selanjutnya dilakukan iterasi *bootstrapping* untuk memperbaiki dataset *training*. Perbaikan terhadap dataset *training* akan digunakan untuk memperbarui model klasifikasi motif batik.



Gambar 3.9 Contoh *ground truth*.

Setelah model klasifikasi motif batik diperbarui, selanjutnya akan dilakukan proses *testing* lagi sampai iterasi *bootstrapping* selesai. Proses *bootstrapping* ini dijelaskan lebih lanjut dalam Subbab 3.2.3. Setelah proses *bootstrapping* selesai, dilakukan pelabelan motif untuk mengetahui jenis motif

yang dideteksi. Pelabelan motif akan dijelaskan lebih lanjut dalam subbab 3.2.4. Selanjutnya dilakukan proses perhitungan akurasi. Perhitungan akurasi ini dilakukan terhadap seluruh data *testing*. Kemudian rata-rata akurasi dihitung dari keseluruhan data *testing*. Proses hitung akurasi ini dijelaskan lebih lanjut dalam subbab 3.2.5. Alur dari proses *testing* ditampilkan dalam *flowchart* pada Gambar 3.10.



Gambar 3.10 *Flowchart* proses *testing*.

3.2.1. Model klasifikasi *multilabel*

Model klasifikasi *multilabel* merupakan kumpulan model klasifikasi motif yang digunakan untuk mendeteksi keberadaan motif dalam citra.

Model klasifikasi *multilabel* dibagi menjadi model-model klasifikasi tiap motif batik. Masing-masing model klasifikasi motif berasal dari *strong classifier* masing-masing motif. Setiap model klasifikasi motif berfungsi untuk mendeteksi motif berdasarkan jenis klasifikasi motif. Model klasifikasi motif parang berfungsi untuk mendeteksi motif parang sedangkan model klasifikasi motif kawung berfungsi untuk mendeteksi motif kawung, begitu juga dengan model klasifikasi motif lainnya yang akan mendeteksi motif berdasarkan jenis klasifikasi motif masing-masing. Setiap model klasifikasi motif ini akan membedakan antara (objek) motif dan *background* pada citra.

Yang dianggap *background* adalah bagian-bagian citra selain motif yang sesuai dengan model klasifikasi motif yang digunakan. Oleh karena itu dalam model klasifikasi motif tertentu, selain motif yang sesuai klasifikasi motif akan dianggap sebagai *background*. Motif yang tidak sesuai model klasifikasi motif juga dianggap sebagai *background*. Misalkan dalam klasifikasi motif parang, yang dianggap sebagai objek motif hanyalah motif parang. Jika terdapat bagian citra yang bukan termasuk motif parang atau motif lain selain motif parang maupun tidak termasuk motif apapun akan dianggap sebagai *background*. Hasil output dari model klasifikasi motif berupa *region-region* citra yang diidentifikasi memiliki motif.

Misalkan digunakan model klasifikasi motif parang. Langkah-langkah yang digunakan untuk melakukan klasifikasi motif parang pada citra *testing* adalah sebagai berikut:

1. Untuk mengidentifikasi posisi motif dalam citra, digunakan *sliding window* dalam citra. *Sliding window* akan digeser dalam citra *testing* dari posisi kiri atas ke kanan bawah. Setiap bagian citra (*region* citra) yang berada dalam *sliding window* akan diidentifikasi keberadaan motifnya. *Sliding window* memiliki ukuran 50x50 piksel sesuai ukuran citra dalam dataset *training*. *Sliding window* digeser dalam citra *testing* secara overlap dengan jarak pergeseran 4 piksel.

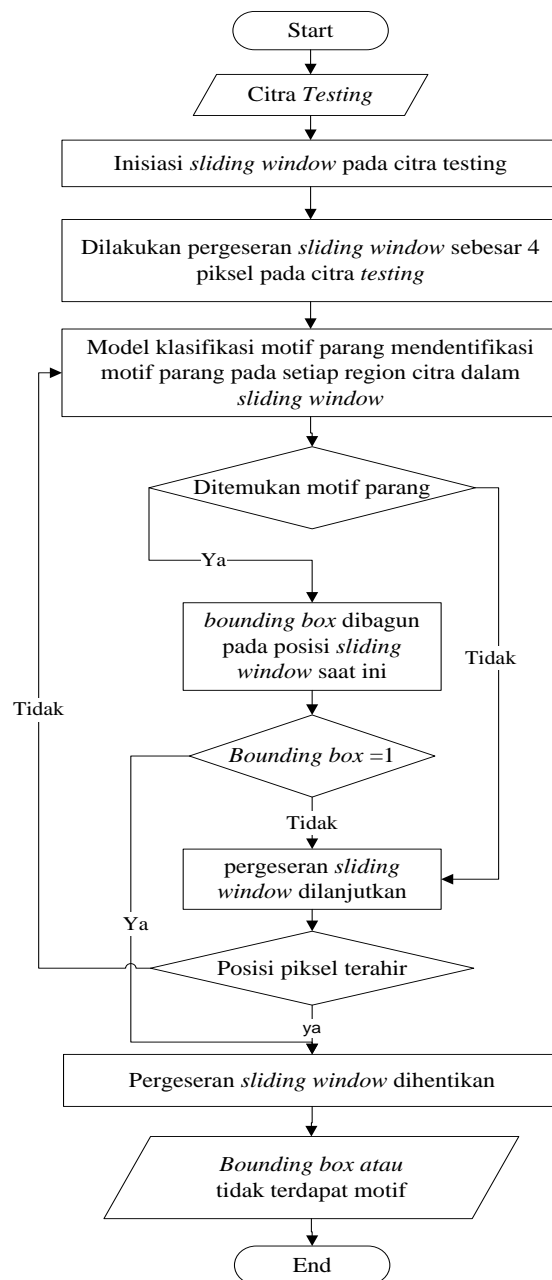
2. Dari setiap *region* citra yang terbentuk oleh *sliding window*, dilakukan klasifikasi motif parang menggunakan model klasifikasi motif parang. Klasifikasi dilakukan dengan cara membagi *region* citra ke dalam beberapa *subset random ferns* yang diskriminan terhadap data *training* negatif dan positif. Dari setiap *subset* pada *region* dilakukan klasifikasi *weak classifier*. Hasil dari setiap *weak classifier* dari *subset* kemudian digunakan untuk menentukan apakah terdapat motif parang pada *region* tersebut.
3. Jika pada *sliding window* ditemukan motif parang, *sliding window* diberi warna kuning, dan posisinya disimpan. *Sliding window* hasil klasifikasi tersebut selanjutnya dianggap sebagai *bounding box* yang akan dicocokkan dengan *bounding box ground truth*.
4. Setelah *sliding window* berada pada posisi kanan bawah, pergeseran dilakukan terhadap citra berikutnya.
5. Terdapat dua kemungkinan *output* dari model klasifikasi motif parang dari setiap citra. Pertama jika terdapat motif parang, *outputnya* berupa *bounding box* yang berisi motif. Kedua, jika tidak terdapat motif parang, maka *outputnya* kosong.

Langkah-langkah klasifikasi motif parang oleh model klasifikasi motif parang juga berlaku untuk klasifikasi model klasifikasi motif lainnya. Alur dari proses klasifikasi motif parang ditampilkan dalam Gambar 3.11.

3.2.2. Pelabelan data *testing*

Dalam melakukan klasifikasi *multilabel* terhadap semua motif batik, hasil deteksi pada *region* citra dataset *testing* akan diberi label *true* positif, *false* positif dan tidak terdeteksi. *Region* citra akan diberi label *true* positif jika *bounding box* hasil deteksi model klasifikasi motif batik sesuai dengan *bounding box ground truth* objek motif batik. *Region* citra akan dilabeli *false* positif jika *bounding box* hasil deteksi model klasifikasi motif batik tidak sesuai dengan *bounding box ground truth*. Sedangkan

jika *bounding box ground truth* tidak dideteksi model klasifikasi motif batik, maka akan dilabeli tidak terdeteksi dan dijadikan sebagai *region* citra dengan label tidak terdeteksi. Label-label tersebut akan digunakan untuk mengupdate dataset menggunakan *bootstrapping* sehingga dataset menjadi lebih dikriminan terhadap objek motif dan *background*.



Gambar 3.11 Flowchart proses klasifikasi motif batik

Selain itu, citra juga dilabeli motif sesuai hasil identifikasi model klasifikasi motif batik. Jika pada citra terdeteksi motif parang dan semen, maka citra akan diberi label motif parang dan semen. Pelabelan motif dilakukan terhadap seluruh dataset *testing*. Label motif hasil deteksi model klasifikasi motif batik akan digunakan dalam proses cek akurasi untuk mengetahui akurasi dari sistem.

3.2.3. *Bootstrapping*

Bootstrapping digunakan untuk melakukan *generate* dataset baru terhadap dataset *training* kemudian menggunakannya dalam proses *training* berikutnya. Melakukan penambahan dataset baru terhadap dataset *training* akan membuat model klasifikasi batik berubah untuk menyesuaikan dataset.

Hal ini dilakukan untuk mengurangi *global error rate* serta untuk membuat *classifier* lebih sesuai dengan dataset *testing*. Dilakukan 6 variasi iterasi *Bootstrapping*, mulai dari tanpa iterasi (iterasi ke-0) sampai iterasi ke-5.

1. *Generate dataset baru*

Dataset baru yang *digenerate* akan ditambahkan pada masing-masing data *training* yang sesuai motifnya. Misalkan terdapat data dengan label tidak terdeteksi parang, maka data tersebut akan ditambahkan ke dalam dataset *training* positif parang. Sedangkan data dengan label *false* positif parang akan ditambahkan ke dalam dataset *training* negatif parang. Penambahan ini juga dilakukan terhadap data dengan label motif lainnya, semua ditambahkan ke dalam dataset motif yang sesuai. Setiap data dengan satu label *false positif* maupun tidak terdeteksi akan membuat data yang dimasukkan ke dalam dataset *training* bertambah sebanyak 10 data.

2. *Update Classifier*

Setelah dataset *training* berubah, maka ketika dilakukan proses *training*, model klasifikasi motif batik yang dibangun akan berubah.

Setiap perubahan model klasifikasi motif batik akan membuatnya semakin sesuai dengan dataset *testing*, sehingga akurasi diharapkan semakin meningkat.

3.2.4. Hitung akurasi

Dalam perhitungan akurasi, digunakan *tanimoto distance* untuk menghitung akurasi label motif sebenarnya dan label motif hasil deteksi model-model klasifikasi motif batik. Label motif sebenarnya diperoleh dari label motif *ground truth* yang dimiliki masing-masing dataset *testing*. Setelah semua akurasi dari keseluruhan dataset *testing* diperoleh, dilakukan perhitungan rata-rata akurasi. Rata-rata akurasi ini akan menjadi *output* terakhir operasi dari sistem klasifikasi *multilabel boosted random ferns* terhadap motif batik.

3.3. Skenario pengujian

Skenario pengujian dilakukan dengan melakukan analisa performa sistem klasifikasi *multilabel boosted random ferns* terhadap variasi jumlah *weak classifier* dan iterasi *bootstrapping*. Dataset *training* yang digunakan mengikuti iterasi *bootstrapping*. Ketika *bootstrapping* berada pada iterasi ke-0, maka data *training* adalah data *training* original yang belum diupdate. Sedangkan ketika *bootstrapping* berada pada iterasi ke-1 dan selanjutnya, maka data *training* menyesuaikan *update* data *training* dari *bootstrapping*. Sedangkan dataset *testing* yang digunakan, dipilih secara random.

Dalam skenario pengujian, terdapat beberapa parameter berikut:

1. jumlah *subset random fern* yang digunakan (5, 10, 15, 20),
2. jumlah *weak classifier* (100, 200, 300, 400),
3. jumlah iterasi *bootstrapping* (0, 1, 2, 3, 4, 5).

Jumlah *subset random ferns* yang digunakan mempengaruhi kualitas fitur *random ferns* dan perhitungan komputasi. Semakin banyak jumlah *subset random ferns* yang digunakan, maka semakin besar kemungkinan mendapatkan fitur *random*

ferns dengan kualitas yang lebih baik namun perhitungan komputasi akan semakin berat. Berdasarkan penelitian Villamizar et.al (2012) yang menggunakan 15 *subset*, akan dilakukan pengujian menggunakan 4 variasi *subset* untuk memperoleh jumlah *subset* yang optimal. Variasi *subset* yang digunakan yakni 5 *subset*, 10 *subset*, 15 *subset* dan 20 *subset*.

Jumlah *weak classifier* mempengaruhi jumlah *weak classifier* t beserta proses *update* sebaran bobot data *training*. Berdasarkan penelitian Villamizar et.al (2012) yang menggunakan jumlah *weak classifier* 300, akan dilakukan pengujian menggunakan 4 variasi jumlah *weak classifier* untuk memperoleh jumlah iterasi yang optimal. Variasi jumlah *weak classifier* yang digunakan yakni 100 iterasi, 200 iterasi, 300 iterasi dan 400 iterasi.

Jumlah iterasi *bootstrapping* mempengaruhi *update* dataset training dan perubahan *strong classifier* yang membentuk model klasifikasi motif. Berdasarkan penelitian Villamizar et.al (2012) yang menggunakan 4 iterasi *bootstrapping*, akan dilakukan pengujian menggunakan 6 variasi iterasi *bootstrapping* untuk memperoleh jumlah iterasi yang optimal. Variasi iterasi *bootstrapping* yakni 0 iterasi, 1 iterasi, 2 iterasi, 3 iterasi, 4 iterasi dan 5 iterasi.

Jumlah skenario pengujian yang dilakukan adalah 96 skenario, di mana masing-masing skenario menggunakan kombinasi jumlah *subset random ferns*, jumlah *weak classifier* dan jumlah iterasi *bootstrapping* yang berbeda. Dalam setiap skenario, performa sistem diukur berdasarkan *tanimoto distance*. Dengan menggunakan *tanimoto distance*, dapat diketahui akurasi label motif hasil identifikasi *classifier* terhadap label motif sebenarnya. Perhitungan akurasi *tanimoto distance* telah diterangkan dalam subbab 2.4.

[Halaman ini sengaja dikosongkan]

BAB 4

HASIL DAN PEMBAHASAN

Pada bab ini dijelaskan tentang hasil pengujian terhadap penelitian yang dilakukan. Selanjutnya dilakukan analisa terhadap hasil pengujian untuk mendapatkan informasi yang akan menjadi kesimpulan penelitian.

4.1. Implementasi Metode

Penelitian diimplementasikan menggunakan perangkat keras dengan spesifikasi prosesor Intel core i3 3.4 GHz dan RAM 4GB. Perangkat lunak yang digunakan adalah *Matlab R2013a* dan sistem operasi *Windows 7*. Dataset *training* menggunakan kombinasi dataset positif dan negatif yang terdiri dari 6 dataset motif batik (parang 40 citra, kawung 40 citra, lunglungan 30 citra, megamendung 30 citra, nitik 50 citra dan semen 30 citra) dan 1 dataset *background* (120 citra). Perbedaan jumlah citra dalam dataset motif batik dikarenakan perbedaan jumlah variasi setiap motif, pada motif parang terdapat 4 variasi, motif kawung 4 variasi, motif lunglungan 3 variasi, motif megamendung 3 variasi, motif nitik 5 variasi dan motif semen 3 variasi. Setiap variasi motif terdiri dari 10 citra.



Gambar 4.1 Contoh data *testing* motif satuan

Sedangkan dataset *testing* yang digunakan terdiri dari 64 citra batik yang terbagi menjadi 42 citra motif satuan dan 22 citra motif kombinasi. Citra motif satuan terdiri dari 6 citra parang, 9 citra kawung, 6 citra lunglungan, 9 citra megamendung, 6 citra semen dan 6 citra nitik. Citra motif kombinasi terdiri dari 2 citra kombinasi parang-kawung-lunglungan, 4 citra kombinasi parang-megamendung, 7 citra kombinasi parang-nitik, 3 citra kombinasi parang-semen, 3 citra kombinasi kawung-nitik dan 3 citra kombinasi semen lunglungan. Contoh citra motif satuan ditampilkan dalam Gambar 4.1. Sedangkan contoh citra motif kombinasi ditampilkan dalam gambar 4.2.



Gambar 4.2 Contoh data testing motif kombinasi

4.2. Hasil Pengujian

Berdasarkan Bab 3, skenario pengujian mencakup kombinasi 4 variasi *subset ferns* dengan 4 variasi jumlah *weak classifier* dan pengujian 6 variasi iterasi *bootstrapping*. Untuk mempermudah pembagian pengujian, dilakukan pengujian kombinasi *subset ferns* dan jumlah *weak classifier* terhadap data *training*. Kemudian dilakukan pengujian kombinasi *subset ferns* dan jumlah *weak classifier* terhadap data *testing*. Terakhir dilakukan pengujian terhadap iterasi *bootstrapping*.

4.2.1. Hasil pengujian model klasifikasi motif terhadap data training

Pengujian ini dilakukan untuk menguji model-model klasifikasi

motif terhadap data *training* (positif dan negatif). Model-model klasifikasi motif akan melakukan klasifikasi terhadap data *training* masing-masing. Kemudian hasil klasifikasi yang benar akan dibagi dengan total jumlah data *training*. Hasil pengujian ini berupa nilai persentase data yang berhasil diklasifikasi benar oleh model-model klasifikasi motif. Hasil pengujian ditampilkan dalam Tabel 4.1, Tabel 4.2, Tabel 4.3 dan Tabel 4.4 untuk masing-masing jumlah *weak classifier* 100, 200, 300, dan 400.

Tabel 4.1 Pengujian akurasi jumlah *weak classifier* 100 pada data *training*

Motif	Jumlah <i>Ferns</i>			
	5	10	15	20
parang	92,61%	94,10%	94,29%	94,66%
kawung	88,97%	90,85%	91,22%	91,69%
lunglungan	84,76%	85,79%	86,18%	86,98%
megamendung	83,64%	84,67%	84,86%	86,01%
semen	83,86%	86,17%	86,34%	88,19%
nitik	79,22%	79,05%	79,53%	81,70%

Tabel 4.2 Pengujian akurasi jumlah *weak classifier* 200 pada data *training*

Motif	Jumlah <i>Ferns</i>			
	5	10	15	20
parang	92,16%	93,93%	94,26%	94,84%
kawung	88,26%	90,36%	91,64%	91,46%
lunglungan	84,03%	85,60%	86,31%	86,98%
megamendung	83,42%	84,70%	84,95%	85,08%
semen	83,03%	85,23%	85,79%	86,67%
nitik	78,11%	78,77%	79,35%	80,58%

Tabel 4.3 Pengujian akurasi jumlah *weak classifier* 300 pada data *training*

Motif	Jumlah <i>Ferns</i>			
	5	10	15	20
parang	92,38%	93,47%	94,53%	94,86%
kawung	88,61%	91,06%	90,92%	92,38%
lunglungan	83,75%	85,96%	86,17%	87,21%
megamendung	83,15%	84,27%	84,97%	85,23%
semen	84,68%	85,84%	85,96%	86,84%
nitik	78,11%	79,03%	79,22%	81,19%

Tabel 4.4 Pengujian akurasi jumlah *weak classifier* 400 pada data *training*

Motif	Jumlah <i>Ferns</i>			
	5	10	15	20
parang	92,65%	93,93%	94,49%	94,82%
kawung	88,58%	90,61%	91,35%	91,59%
lunglungan	84,58%	85,73%	86,22%	86,83%
megamendung	83,10%	84,47%	84,95%	85,19%
semen	84,24%	85,53%	85,99%	87,10%
nitik	77,72%	79,48%	80,05%	80,98%

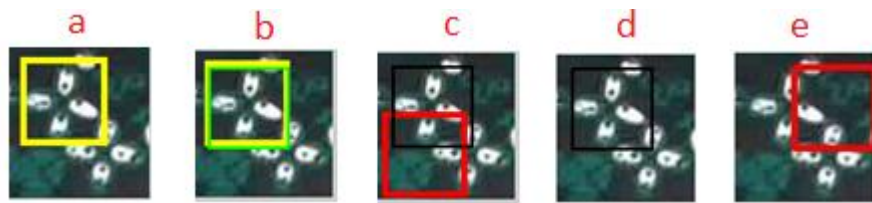
Pada Tabel 4.1, Tabel 4.2, Tabel 4.3 dan Tabel 4.4, nilai akurasi tertinggi diperoleh dari motif parang dengan akurasi 94,66% (jumlah *weak classifier* 100), 94,84% (jumlah *weak classifier* 200), 94,86% (jumlah *weak classifier* 300) dan 94,82% (jumlah *weak classifier* 400). Sedangkan akurasi terendah diperoleh dari motif nitik dengan akurasi 79,5%(iterasi 100), 78,11% (jumlah *weak classifier* 200 dan 300) dan 77,72%(jumlah *weak classifier* 400). Untuk masing-masing motif, nilai akurasi tertinggi selalu berada pada *subset ferns* 20, baik pada iterasi 100, 200, 300 maupun 400 *weak classifier*.

Nilai akurasi tertinggi dari motif parang adalah 94,86% (*subset ferns* 20 dengan jumlah *weak classifier* 300). Nilai akurasi tertinggi dari motif kawung adalah 92,38% (*subset ferns* 20 dengan jumlah *weak classifier* 300). Nilai akurasi tertinggi dari motif lunglungan adalah 87,21% (*subset ferns* 20 dengan jumlah *weak classifier* 300). Nilai akurasi tertinggi dari motif megamendung adalah 86,01% (*subset ferns* 20 dengan jumlah *weak classifier* 100). Nilai akurasi tertinggi dari motif semen adalah 88,19% (*subset ferns* 20 dengan jumlah *weak classifier* 100). Nilai akurasi tertinggi dari motif nitik adalah 81,70% (*subset ferns* 20 dengan jumlah *weak classifier* 100).

4.2.2. Hasil pengujian model klasifikasi motif terhadap data *testing*

Selanjutnya model-model klasifikasi motif melakukan pengujian terhadap data *testing*. Hasil dari pengujian ini adalah label motif dari

masing-masing data *testing*. Untuk menghitung akurasi multilabel dari data *testing* digunakan *tanimoto distance*. Pada *tanimoto distance*, semakin mendekati 0, maka nilai persamaan semakin tinggi. Pengujian terhadap data *testing* dilakukan untuk mengetahui jumlah rata-rata nilai *tanimoto distance*, jumlah file yang dilabeli secara benar, jumlah file yang dilabeli secara salah, waktu *testing* dan waktu *training*.






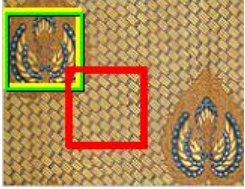


Gambar 4.3 Ilustrasi hasil deteksi model klasifikasi motif. (a) Data hasil deteksi model klasifikasi motif. (b) Data hasil deteksi model klasifikasi motif sesuai *ground truth*. (c) Data hasil deteksi model klasifikasi motif tidak sesuai *ground truth*. (d) Data *ground truth* tanpa hasil klasifikasi (tidak terdeteksi). (e) Data hasil deteksi model klasifikasi motif tanpa *ground truth*.

Dalam pengujian terhadap data *testing*, model-model klasifikasi akan mencari objek motif dalam setiap citra testing. Setiap menemukan objek yang dianggap sebagai motif, model klasifikasi akan menandai dengan *bounding box* berwarna kuning seperti pada Gambar 4.3 bagian a. Selain memberi tanda, model klasifikasi juga akan sekaligus memberi label motif pada citra yang dianggap memiliki objek motif tersebut. Pelabelan ini berarti jika data tidak sesuai dengan *ground truth* (*false positif*) maka citra tetap dilabeli sesuai hasil klasifikasi model klasifikasi motif.

Jika hasil klasifikasi sesuai dengan *ground truth*, maka hasil klasifikasi akan ditindas dengan *bounding box* *ground truth* berwarna hijau seperti pada Gambar 4.3 bagian b. Namun jika tidak sesuai *ground truth* (*false positif*), maka *bounding box* *ground truth* akan berwarna hitam dan *bounding box* hasil klasifikasi akan berwarna merah seperti pada Gambar

4.3 bagian c. Sedangkan jika hanya ada ground truth saja tanpa ada hasil klasifikasi seperti pada Gambar 4.3 bagian d maka bounding box ground truth juga akan berwarna hitam dan citra tidak diberi label motif. Dan jika hanya ada hasil klasifikasi tanpa ada *ground truth* (klasifikasi salah mendeteksi *background* sebagai objek motif) seperti pada Gambar 4.3 bagian e, maka bounding box hasil klasifikasi akan berwarna merah namun citra tetap diberi label motif sesuai model klasifikasi motif. Contoh pelabelan hasil deteksi motif ditampilkan dalam Tabel 4.5. Untuk contoh lengkap disajikan dalam lembar lampiran.

Tabel 4.5 Contoh hasil pelabelan deteksi motif

No	Hasil Testing	Label hasil klasifikasi	Label sebenarnya	Nilai <i>tanimoto distance</i>
1		kawung	kawung	0
2		parang-nitik	parang-nitik	0
3		-	Semen	1
4		semen-nitik	Semen	0.5
5		semen	Semen	0
6		kawung-nitik	Kawung	0.5

Kemudian, hasil pengujian terhadap data *testing* ditampilkan dalam Tabel 4.6, Tabel 4.7, Tabel 4.8 dan Tabel 4.9 untuk masing-masing pengujian dengan jumlah *weak classifier* 100, 200, 300 dan 400.

Tabel 4.6 Pengujian jumlah *weak classifier* 100 pada data *testing*

Jumlah <i>Ferns</i>	Rata-rata nilai <i>tanimoto</i>	Jumlah file benar	Jumlah file salah	Waktu testing (detik)	Waktu training (detik)
5	0.0677	57	7	1788	121
10	0.0820	55	9	1762	156
15	0.0729	55	9	1772	191
20	0.0599	59	5	1771	225

Tabel 4.7 Pengujian jumlah *weak classifier* 200 pada data *testing*

Jumlah <i>Ferns</i>	Rata-rata nilai <i>tanimoto</i>	Jumlah file benar	Jumlah file salah	Waktu testing (detik)	Waktu training (detik)
5	0.0469	60	4	3545	245
10	0.0599	58	6	3778	341
15	0.0521	59	5	3814	410
20	0.0599	58	6	3495	480

Tabel 4.8 Pengujian jumlah *weak classifier* 300 pada data *testing*

Jumlah <i>Ferns</i>	Rata-rata nilai <i>tanimoto</i>	Jumlah file benar	Jumlah file salah	Waktu testing (detik)	Waktu training (detik)
5	0.0469	60	4	5217	332
10	0.0677	58	6	5181	403
15	0.0469	60	4	5489	518
20	0.0521	59	5	5470	628

Tabel 4.9 Pengujian jumlah *weak classifier* 400 pada data *testing*

Jumlah <i>Ferns</i>	Rata-rata nilai <i>tanimoto</i>	Jumlah file benar	Jumlah file salah	Waktu testing (detik)	Waktu training (detik)
5	0.0521	59	5	8586	395
10	0.0547	59	5	8332	597
15	0.0547	59	5	7894	740
20	0.0599	59	5	6663	866

Dari Tabel 4.5, Tabel 4.6, Tabel 4.7 dan Tabel 4.8, dapat diketahui bahwa nilai *tanimoto distance* terbaik terdapat dalam jumlah *weak classifier* 200 (5 *ferns*) dan 300 (5 dan 15 *ferns*) dengan nilai 0.0469 dengan jumlah file benar sebanyak 60 dan file yang salah hanya 4. Dari segi waktu, semakin banyak jumlah *ferns* dan semakin banyak jumlah *weak classifier* maka semakin lama waktu *training* yang dibutuhkan. Sedangkan untuk waktu *testing*, semakin semakin banyak jumlah *weak classifier* maka semakin lama waktu pengujian *testing* yang di butuhkan.

Untuk mengetahui hasil rincian pengujian, pada Tabel 4.10 dan Tabel 4.11 ditampilkan rincian pengujian terhadap data *testing* menggunakan 5 *subset ferns* dan jumlah *weak classifier* 100.

Tabel 4.10 Pengujian terhadap data *testing* (5 *ferns* dengan jumlah *weak classifier* 100)

No	Nama file	Hasil pengujian model klasifikasi (1= ditemukan objek, 0=background)					
		Parang	Kawung	Lung- Lungan	Mega- mendung	Semen	Nitik
1	kawung_1_1	0	1	0	0	0	0
2	kawung_1_2	0	1	0	0	0	0
3	kawung_1_3	0	1	0	0	0	0
4	kawung_2_1	0	1	0	0	0	0
5	kawung_2_2	0	1	0	0	0	0
6	kawung_2_3	0	1	0	0	0	0
7	kawung_3_1	0	1	0	0	0	0
8	kawung_3_2	0	1	0	0	0	1
9	kawung_3_3	0	1	0	0	0	1
10	kawung_nitik_1_1	0	1	0	0	0	0
11	kawung_nitik_1_2	0	1	0	0	0	1
12	kawung_nitik_1_3	0	1	0	0	0	1
13	lunglungan_1_1	0	0	1	0	0	0
14	lunglungan_1_2	0	0	1	0	0	0
15	lunglungan_1_3	0	0	1	0	0	0
16	lunglungan_2_1	0	0	1	0	0	0
17	lunglungan_2_2	0	0	1	0	0	0
18	lunglungan_2_3	0	0	1	0	0	0

Tabel 4.11 Lanjutan Tabel 4.10

No	Nama file	Hasil pengujian model klasifikasi (1= ditemukan objek, 0=background)					
		Parang	Kawung	Lung- Lungan	Mega- mendung	Semen	Nitik
19	megamendung_1-1	0	0	0	1	0	0
20	megamendung_1-2	0	0	0	1	0	0
21	megamendung_1-3	0	0	0	1	0	0
22	megamendung_2-1	0	0	0	1	0	0
23	megamendung_2-2	0	0	0	1	0	0
24	megamendung_2-3	0	0	0	1	0	0
25	megamendung_3-1	0	0	0	1	0	0
26	megamendung_3-2	0	0	0	1	0	0
27	megamendung_3-3	0	0	0	1	0	0
28	nitik_1_1	0	0	0	0	0	1
29	nitik_1_2	0	0	0	0	0	1
30	nitik_1_3	0	0	0	0	0	0
31	nitik_2_1	0	0	0	0	0	1
32	nitik_2_2	0	0	0	0	0	1
33	nitik_2_3	0	0	0	0	0	1
34	parang_1_1	1	0	0	0	0	0
35	parang_1_2	1	0	0	0	0	0
36	parang_1_3	1	0	0	0	0	0
37	parang_2_1	1	0	0	0	0	0
38	parang_2_2	1	0	0	0	0	0
39	parang_2_3	1	0	0	0	0	0
40	parang_kawung_lunglungan_1_1	1	1	0	0	0	0
41	parang_kawung_lunglungan_1_2	1	1	1	0	0	0
42	parang_megamendung_1_1	1	0	0	1	0	0
43	parang_megamendung_1_2	1	0	0	1	0	0
44	parang_megamendung_1_3	1	0	0	1	0	0
45	parang_megamendung_1_4	1	0	0	1	0	0
46	parang_nitik_1_1	1	0	0	0	0	1
47	parang_nitik_1_2	1	0	0	0	0	1

Tabel 4.12 Lanjutan Tabel 4.11

No	Nama file	Hasil pengujian model klasifikasi (1= ditemukan objek, 0=background)					
		Parang	Kawung	Lung- Lungan	Mega- mendung	Semen	Nitik
48	parang_nitik_1_3	1	0	0	0	0	1
49	parang_nitik_2_1	1	0	0	0	0	1
50	parang_nitik_2_2	1	0	0	0	0	1
51	parang_nitik_2_3	1	0	0	0	0	1
52	parang_nitik_2_4	1	0	0	0	0	1
53	parang_semen_1-1	1	0	0	0	1	0
54	parang_semen_1-2	1	0	0	0	1	0
55	parang_semen_1-3	1	0	0	0	1	0
56	semen_1-1	0	0	0	0	1	0
57	semen_1-2	0	0	0	0	0	0
58	semen_2-1	0	0	0	0	1	1
59	semen_2-2	0	0	0	0	1	0
60	semen_2-3	0	0	0	0	1	0
61	semen_2_4	0	0	0	0	1	0
62	semen_lunglungan_1_1	0	0	1	0	1	0
63	semen_lunglungan_1_2	0	0	1	0	1	0
64	semen_lunglungan_1_3	0	0	1	0	1	0

Pada Tabel 4.10 , 4.11 dan 4.12, citra yang diklasifikasi secara benar oleh model klasifikasi ditandai dengan warna hijau. Sedangkan jika terdapat *background* yang diklasifikasikan sebagai objek oleh model klasifikasi, akan diberi tanda warna kuning. Jika terdapat objek yang tidak terdeteksi oleh model klasifikasi maka diberi warna merah.

Jumlah objek motif yang diklasifikasi secara benar oleh model klasifikasi (warna hijau) sebanyak 84 objek. Jumlah *background* yang diklasifikasi sebagai objek oleh model klasifikasi (warna kuning) ada 3 *background*. Sedangkan objek yang tidak terklasifikasi oleh model klasifikasi (warna merah) ada 4 objek. Dari segi jumlah file, hanya

terdapat 7 file yang diklasifikasi salah oleh model klasifikasi motif, sedangkan 57 lainnya diklasifikasikan secara benar oleh model klasifikasi motif. Hasil tampilan dari pengujian data *testing* menggunakan 5 *subset ferns* dan 100 jumlah *weak classifier* pada Tabel 4.5 disajikan dalam lembar lampiran.

4.2.3. Hasil pengujian *bootstrapping*

Selanjutnya dilakukan pengujian *bootstrapping*. Sama seperti pengujian data *testing*, hasil dari pengujian *bootstrapping* adalah label motif dari masing-masing data *testing*. Digunakan 6 variasi iterasi *bootstrapping* yang diujikan, yakni *bootstrapping* ke-0 (tanpa *bootstrapping*) dan iterasi *bootstrapping* ke-1 sampai ke-5. Sama seperti pengujian terhadap data *testing*, pengujian *bootstrapping* dilakukan untuk mengetahui jumlah rata-rata nilai *tanimoto distance*, jumlah file yang dilabeli secara benar, jumlah file yang dilabeli secara salah, waktu *testing* dan waktu *training*..

Digunakan kombinasi jumlah *subset ferns* dan jumlah *weak classifier* yang sama dalam pengujian *bootstrapping* yakni 5 *subset ferns* dengan 100 jumlah *weak classifier*. Kombinasi jumlah *subset ferns* dan jumlah *weak classifier* tersebut selanjutnya digunakan dalam pengujian terhadap iterasi *bootstrapping* ke-0 sampai ke-5. Hasil dari pengujian *bootstrapping* ditampilkan dalam Tabel 4.13.

Tabel 4.13 Pengujian *Bootstrapping*

Iterasi <i>bootstrapping</i>	Rata-rata nilai <i>tanimoto</i>	Jumlah file benar	Jumlah file salah	Waktu testing (detik)	Waktu training (detik)
0	0.0677	57	7	1788	121
1	0.0521	57	7	2010	119
2	0.0130	62	2	2025	128
3	0.0469	59	5	2032	135
4	0.0469	60	4	2033	144
5	0.0234	61	3	2010	148

Pada Tabel 4.13, nilai *tanimoto distance* terbaik ada pada iterasi *bootstrapping* ke-2 dengan nilai 0.0130 dan jumlah file benar 62 dan file salah 2. Waktu *training* juga mengalami kenaikan semakin banyak iterasi *bootstrapping* dilakukan, hal ini karena jumlah data *training* juga bertambah seiring bertambahnya iterasi *bootstrapping*. Untuk waktu *testing* ketika sudah memasuki iterasi *bootstrapping*, cenderung stabil pada kisaran 2000 detik karena jumlah *subset ferns* dan jumlah *weak classifier* tidak berubah.

4.3. Pembahasan Hasil Pengujian

Dari pengujian *training* yang ditampilkan pada Tabel 4.1, Tabel 4.2, Tabel 4.3 dan Tabel 4.4, sebagian besar data menunjukkan bahwa semakin banyak *ferns* yang digunakan maka semakin besar akurasi terhadap data training. Bahkan nilai akurasi terbesar masing-masing motif terdapat pada *subset ferns* 20 pada setiap iterasi. Hal ini dikarenakan pada pengujian training, jumlah *subset ferns* masih berpengaruh terhadap pemilihan *subset ferns* terbaik (melalui *bhattacharya distance*). Sedangkan pada pengujian terhadap citra *testing*, pilihan *subset ferns* terbaik untuk data training belum tentu menjadi yang terbaik untuk data *testing*.

Namun untuk jumlah *weak classifier*, terdapat beberapa variasi peningkatan akurasi. Akurasi pada jumlah *weak classifier* yang lebih banyak terkadang memiliki akurasi yang lebih rendah daripada akurasi *weak classifier* yang lebih sedikit. Berdasarkan hal tersebut dapat disimpulkan bahwa pada pengujian *training*, jumlah *subset ferns* memiliki pengaruh peningkatan akurasi yang stabil daripada jumlah *weak classifier*.

Selanjutnya dalam pengujian data *testing* yang ditampilkan dalam Tabel 4.5, Tabel 4.6, Tabel 4.7 dan Tabel 4.8, nilai *tanimoto distance* cenderung meningkat dari jumlah *weak classifier* 100 sampai 300. Hanya saja pada jumlah *weak classifier* 400 nilai *tanimoto distance* menurun menjadi lebih rendah dari jumlah *weak classifier* 300. Hanya pada *subset ferns* 10 saja nilai *tanimoto distance* yang meningkat dari 0.677 menjadi 0.547, sedangkan pada *subset ferns* 5

dan 15 menurun dari 0.0469 menjadi 0.0521 dan 0.0547. Pada pengujian data *testing*, jumlah *subset ferns* tidak terlalu berpengaruh dalam perolehan nilai *tanimoto distance* terbaik. Hal ini karena *subset ferns* yang dipilih untuk model klasifikasi diperoleh secara random dan dipilih satu yang memiliki nilai *bhattacharya* terendah baik itu dari 5 *ferns*, 10 *ferns*, 15 *ferns* maupun 20 *ferns*.

Dari segi waktu, semakin banyak jumlah *ferns* dan semakin banyak jumlah *weak classifier* maka semakin lama waktu *training* yang dibutuhkan. Hal tersebut karena komputasi pembuatan model dalam proses *training* menjadi semakin banyak, berbanding lurus dengan semakin banyaknya *subset ferns* dan *weak classifier* yang digunakan. Sedangkan untuk waktu *testing*, semakin banyak jumlah *weak classifier* maka semakin lama waktu pengujian *testing* yang dibutuhkan. Akan tetapi dalam jumlah jumlah *weak classifier* yang sama waktu yang dibutuhkan untuk melakukan *testing* cenderung stabil, karena jumlah *ferns* yang digunakan dalam *testing* sama (dari masing-masing variasi jumlah *subset ferns* hanya diambil satu yang memiliki nilai *bhattacharya* terendah).

Pada pengujian *bootstrapping*, berdasarkan Tabel 4.13 terjadi peningkatan nilai *tanimoto distance* dari iterasi *bootstrapping* ke-0 sampai ke-2, namun pada iterasi ke-3 nilai *tanimoto distance* kembali turun dan kemudian meningkat kembali sampai iterasi ke-5. Dalam pengujian *bootstrapping*, terdapat 2 nilai *tanimoto distance* yang lebih baik dari nilai *tanimoto distance* terbaik dari pengujian data *testing*. Nilai *tanimoto distance* terbaik dari pengujian data *testing* adalah 0.469 sedangkan pada iterasi *bootstrapping* ke-2 dan ke-5 nilai *tanimoto distance* adalah 0.0130 dan 0.0234. Hal tersebut dikarenakan setelah mengalami proses *bootstrapping*, dataset yang digunakan diupdate sehingga memperoleh model klasifikasi yang lebih baik daripada tanpa proses *bootstrapping*.

Sedangkan dari segi waktu, secara umum terjadi pertambahan waktu *training* seiring bertambahnya iterasi *bootstrapping*. Pertambahan waktu *training* dikarenakan pertambahan jumlah citra dalam dataset *training* dari setiap iterasi *bootstrapping* yang berasal dari data citra dengan label *false positif* dan tidak terdeteksi. Sehingga semakin banyak jumlah dataset *training*, maka semakin lama

pula waktu *training* yang diperlukan. Kemudian dalam waktu *testing*, hanya terdapat perbedaan signifikan pada waktu iterasi ke-0 dibanding iterasi lainnya, hal ini hanya anomali perubahan waktu saja karena mulai iterasi *bootstrapping* ke-1 dan seterusnya, waktu *testing* cenderung stabil. Stabilitasnya waktu *testing* ini dikarenakan tidak ada perubahan jumlah *subset ferns* maupun jumlah *weak classifier* (menggunakan jumlah yang sama).

BAB 5

PENUTUP

Pada bab ini dijelaskan tentang kesimpulan dan saran dari serangkaian pengujian pada bab sebelumnya.

5.1. Kesimpulan

Berdasarkan penelitian ini, kesimpulan yang dapat diambil yaitu:

- a. Pada pengujian *training*, jumlah *subset ferns* memiliki pengaruh peningkatan akurasi yang stabil daripada jumlah *weak classifier*. Jumlah akurasi terbesar diperoleh dari motif parang pada jumlah *weak classifier* 300 dengan *subset ferns* 20 yakni sebesar 94.86%. sedangkan akurasi terendah diperoleh dari motif titik pada jumlah *weak classifier* 400 dengan *subset ferns* 20 77.72%.
- b. Pada pengujian testing, digunakan tanimoto distance untuk pengujian multilabel. Nilai tanimoto distance terbaik diperoleh dari *subset ferns* 5 pada jumlah *weak classifier* 200 serta *subset ferns* 5 dan *subset ferns* 15 pada jumlah *weak classifier* 300 dengan nilai yang sama, yakni 0.469. Dari segi waktu, semakin banyak jumlah *subset ferns* maupun *weak classifier* yang digunakan, maka waktu *training* dan *testing* yang dibutuhkan juga semakin banyak karena beban komputasi juga bertambah. Namun dalam jumlah *weak classifier* yang sama, waktu *testing* dengan variasi *subset ferns* cenderung stabil.
- c. Pada pengujian *bootstrapping*, diperoleh dua nilai tanimoto distance yang lebih baik dari pengujian testing yakni dari iterasi *bootstrapping* ke-2 dan ke-5 yang masing-masing bernilai 0.0130 dan 0.0234. *Bootstrapping* berpengaruh pada perubahan dataset, sehingga semakin baik dataset yang digunakan akan semakin baik model klasifikasi yang dihasilkan. Jumlah iterasi *bootstrapping* mempengaruhi jumlah dataset sehingga waktu *training* semakin bertambah seiring bertambahnya jumlah iterasi *bootstrapping*. Sedangkan untuk waktu *testing* cenderung stabil selama menggunakan jumlah *ferns* dan *weak classifier* yang sama.

5.2. Saran

Saran yang dapat diberikan untuk pengembangan lebih lanjut penelitian ini adalah:

- a. Penelitian dapat ditingkatkan performanya dengan menggunakan dataset yang memiliki variasi bentuk yang tidak terlalu kontas satu

sama lain dalam satu kelas. Hal ini dibuktikan dengan motif nitik yang memiliki bentuk kontras pada variannya membuatnya mendapatkan akurasi terendah dalam pengujian *training*.

- b. Diperlukan penambahan jenis motif batik agar dapat mengklasikan lebih banyak jenis motif batik.




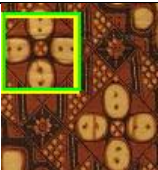



DAFTAR PUSTAKA

- Asiri, N. M., AlHumaidi, N., & AIOsaim, N. (2015). Combination of Histogram of Oriented Gradients and Hierarchical Centroid for Sketch-Based Image Retrieval. *2015 Second International Conference on Computer Science, Computer Engineering, and Social Media (CSCESM)*, 149–152.
- Dalal, N., & Triggs, W. (2005). Histograms of Oriented Gradients for Human Detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR05*, 1(3), 886–893.
- Dhian E K R, I., & Nugraha, K. A. (2016). Klasifikasi batik menggunakan knn berbasis wavelet, *2016* (Sentika), 18–19.
- Kurniawardhani, A., Suciati, N., & Arieshanti, I. (2014). Klasifikasi Citra Batik Menggunakan Metode Ekstraksi Ciri yang Invariant Terhadap Rotasi. *JUTI: Jurnal Ilmiah Teknologi Informasi*, 12(2), 48.
- Nilogiri, A., Suciati, N., & Purwitasari, D. (2012). KLASIFIKASI KANSEI MULTI LABEL DENGAN PROBABILISTIC NEURAL NETWORK PADA CITRA BATIK MENGGUNAKAN. *Seminar Nasional Manajemen Teknologi*, XVI, 1–9.
- Oliveira, E., Ciarelli, P. M., Badue, C., Goiabeiras, C. De, Ferrari, A. F., & Postal, C. (2008). a KNN based Approach and a PNN Algorithm for a Multi-Label Classification Problem, Eighth International Conference on Intelligent Systems Design and Applications.
- Pratikaningtyas, D., Santoso, I., & Ajulian Z., A. (2010). KLASIFIKASI MOTIF BATIK MENGGUNAKAN METODE TRANSFORMASI PAKET WAVELET. *Tugas Akhir*.
- Randa, A. F., Suciati, N., & Navastara, D. A. (2016). Implementasi Metode Kombinasi Histogram Of Oriented Gradients Dan Hierarchical Centroid Untuk Sketch Based Image Retrieval. *JURNAL TEKNIK ITS*, 5(2).
- Rangkuti, A. H. (2013). Klasifikasi Motif Batik Berbasis Kemiripan Ciri Dengan Wavelet Transform Dan Fuzzy Neural Network, (9), 361–372.
- Riesmala, C. P., Rizal, A., & Novamizanti, L. (2012). Pengenalan Motif Batik









- Dengan Analisis Struktur Dan Warna Pada Citra Digital. *Tugas Akhir*, 0–6.
- Sulistyo A.S., P. (2016). Sistem Pengenalan Pola Motif Batik Pada Perangkat Android Dengan Jaringan Syaraf Tiruan, Universitas Gadjah Mada, Yogyakarta..
- Tomasi, C. (2012). Histograms of Oriented Gradients. *Computer Vision Sampler*, 1–6.
- Villamizar, M., Andrade-cetto, J., & Sanfeliu, A. (2017). Boosted Random *Ferns* for Object Detection, 8828(c).
- Villamizar, M., Andrade-Cetto, J., Sanfeliu, A., & Moreno-Noguer, F. (2012). Bootstrapping Boosted Random *Ferns* for discriminative and efficient object classification. *Pattern Recognition*, 45(9), 3141–3153.
- Villamizar, M., Moreno-Noguer, F., Andrade-Cetto, J., & Sanfeliu, A. (2010). Shared random *ferns* for efficient detection of multiple categories. *Proceedings - International Conference on Pattern Recognition*, 388–391.
- www.unesco.org. (2009). Indonesian Batik.

Lampiran A. Hasil Uji Coba






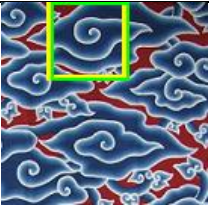
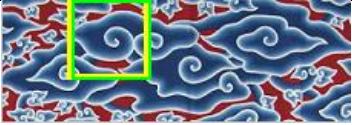
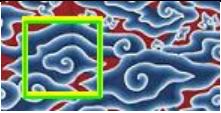
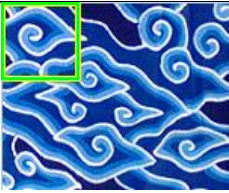
Tabel Hasil Pengujian 5 *Ferns* Dengan 100 *Weak Classifier*

no	Hasil Testing	Label hasil klasifikasi	Label sebenarnya	Nilai <i>tanimoto</i> <i>distance</i>
1		kawung	kawung	0
2		kawung	kawung	0
3		kawung	kawung	0
4		kawung	kawung	0
5		kawung	kawung	0
6		kawung	kawung	0
7		kawung	kawung	0
8		kawung- nitik	kawung	0.5

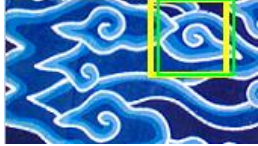

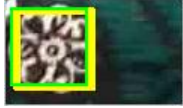

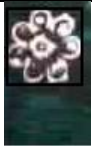
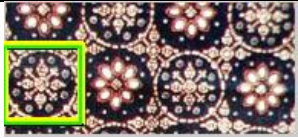
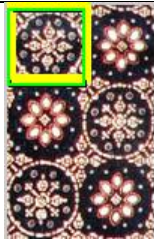

Tabel Lanjutan Tabel Hasil Pengujian 5 *Ferns* Dengan 100 *Weak Classifier*

no	Hasil Testing	Label hasil klasifikasi	Label sebenarnya	Nilai <i>tanimoto</i> <i>distance</i>
9		kawung- nitik	kawung	0.5
10		kawung	kawung- nitik	0.5
11		kawung- nitik	kawung- nitik	0
12		kawung- nitik	kawung- nitik	0
13		lunglungan	lunglungan	0
14		lunglungan	lunglungan	0
15		lunglungan	lunglungan	0
16		lunglungan	lunglungan	0






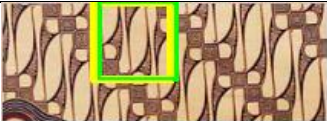
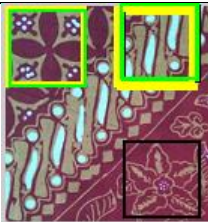
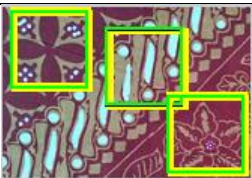
Tabel Lanjutan Tabel Hasil Pengujian 5 *Ferns* Dengan 100 *Weak Classifier*

no	Hasil Testing	Label hasil klasifikasi	Label sebenarnya	Nilai <i>tanimoto</i> <i>distance</i>
17		lunglungan	lunglungan	0
18		lunglungan	lunglungan	0
19		megamendu ng	megamend ung	0
20		megamendu ng	megamend ung	0
21		megamendu ng	megamend ung	0
22		megamendu ng	megamend ung	0
23		megamendu ng	megamend ung	0
24		megamendu ng	megamend ung	0
25		megamendu ng	megamend ung	0

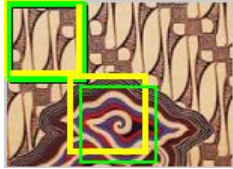
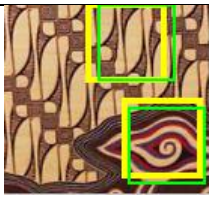
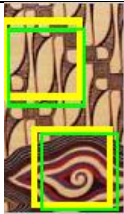

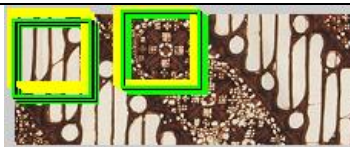


Tabel Lanjutan Tabel Hasil Pengujian 5 *Ferns* Dengan 100 *Weak Classifier*

no	Hasil Testing	Label hasil klasifikasi	Label sebenarnya	Nilai <i>tanimoto</i> <i>distance</i>
26		megamendu ng	megamend ung	0
27		megamendu ng	megamend ung	0
28		nitik	nitik	0
29		nitik	nitik	0
30		-	nitik	1
31		nitik	nitik	0
32		nitik	nitik	0
33		nitik	nitik	0


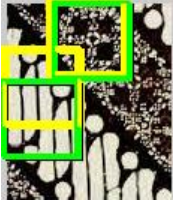


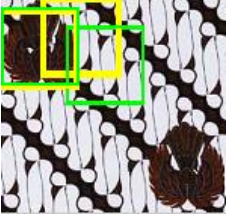

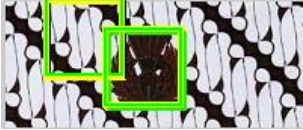


Tabel Lanjutan Tabel Hasil Pengujian 5 *Ferns* Dengan 100 *Weak Classifier*

No	Hasil Testing	Label hasil klasifikasi	Label sebenarnya	Nilai <i>tanimoto</i> <i>distance</i>
34		parang	parang	0
35		parang	parang	0
36		parang	parang	0
37		parang	parang	0
38		parang	parang	0
39		parang	parang	0
40		parang- kawung	parang- kawung- lunglungan	0.333
41		parang- kawung- lunglungan	parang- kawung- lunglungan	0

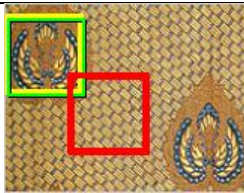




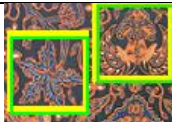
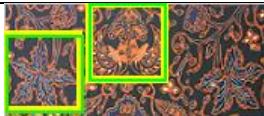
Tabel Lanjutan Tabel Hasil Pengujian 5 *Ferns* Dengan 100 *Weak Classifier*

no	Hasil Testing	Label hasil klasifikasi	Label sebenarnya	Nilai <i>tanimoto</i> <i>distance</i>
42		parang- megamendu ng	parang- megamend ung	0
43		parang- megamendu ng	parang- megamend ung	0
44		parang- megamendu ng	parang- megamend ung	0
45		parang- megamendu ng	parang- megamend ung	0
46		parang-nitik	parang- nitik	0
47		parang-nitik	parang- nitik	0
48		parang-nitik	parang- nitik	0

Tabel Lanjutan Tabel Hasil Pengujian 5 *Ferns* Dengan 100 *Weak Classifier*

No	Hasil Testing	Label hasil klasifikasi	Label sebenarnya	Nilai <i>tanimoto</i> <i>distance</i>
49		parang-nitik	parang-nitik	0
50		parang-nitik	parang-nitik	0
51		parang-nitik	parang-nitik	0
52		parang-nitik	parang-nitik	0
53		parang-semen	parang-semen	0
54		parang-semen	parang-semen	0
55		parang-semen	parang-semen	0
56		semen	semen	0
57		-	semen	1

Tabel Lanjutan Tabel Hasil Pengujian 5 *Ferns* Dengan 100 *Weak Classifier*

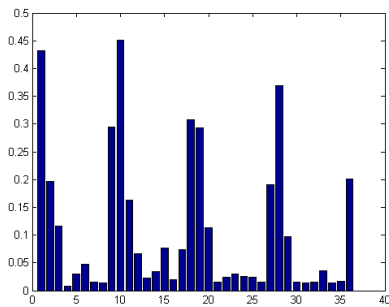
no	Hasil Testing	Label hasil klasifikasi	Label sebenarnya	Nilai <i>tanimoto</i> <i>distance</i>
58		semen-nitik	semen	0.5
59		semen	semen	0
60		semen	semen	0
61		semen	semen	0
62		semen- lunglungan	semen- lunglungan	0
63		semen- lunglungan	semen- lunglungan	0
64		semen- lunglungan	semen- lunglungan	0

Lampiran B. Tampilan Histogram of Oriented Gradient

Berikut ini ditampilkan histogram of oriented gradient (HOG) dari posisi dengan akurasi terbaik dari setiap model klasifikasi motif dengan jumlah subset ferns 5 dan jumlah weak classifier 100.

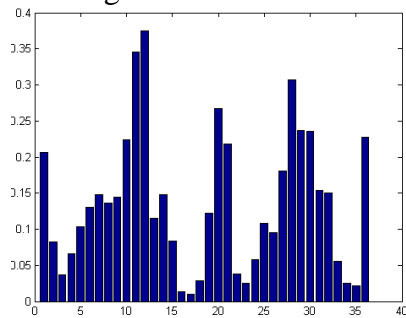
HOG model klasifikasi motif parang

Parang (Objek)

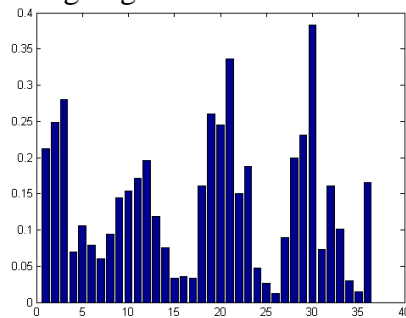


Non Parang (Background)

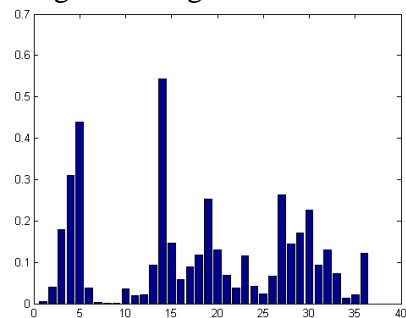
Kawung



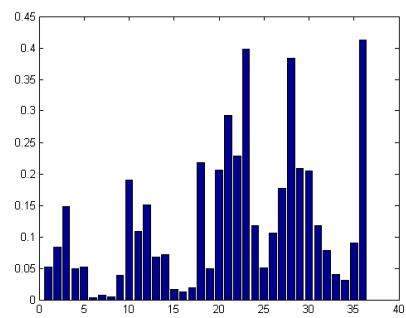
Lunglungan



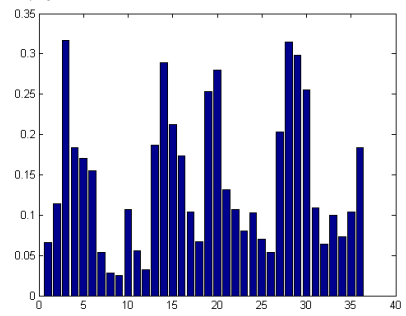
Megamendung



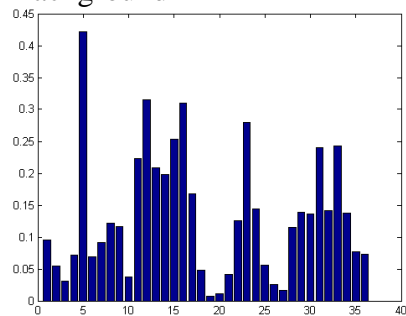
Semen



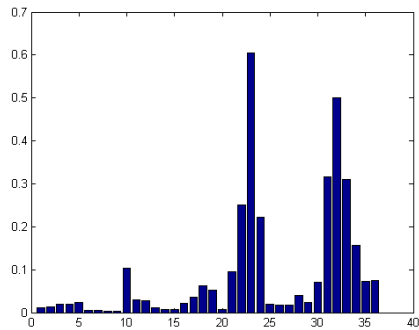
Nitik



Background

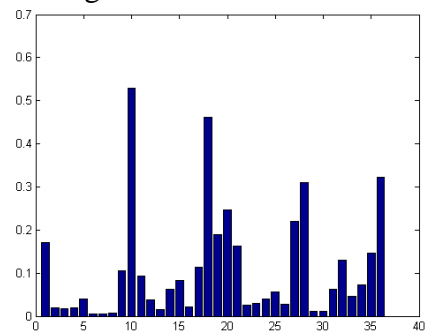


HOG model klasifikasi motif kawung Kawung (Objek)

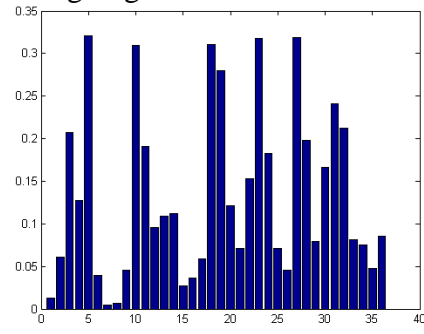


Non Kawung (Background)

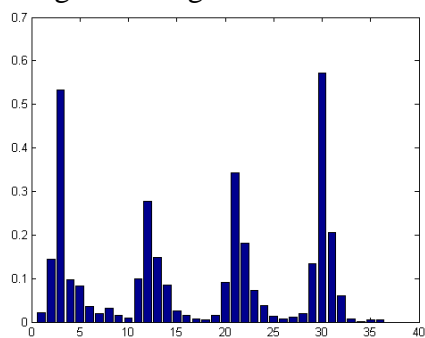
Parang



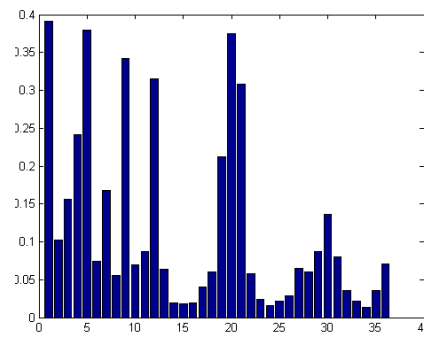
Lunglungan



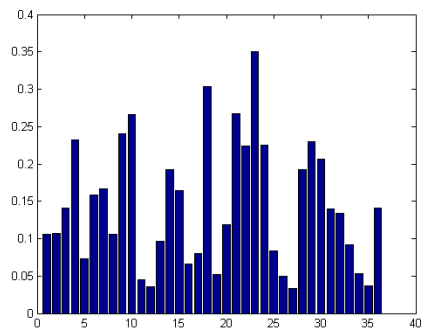
Megamendung



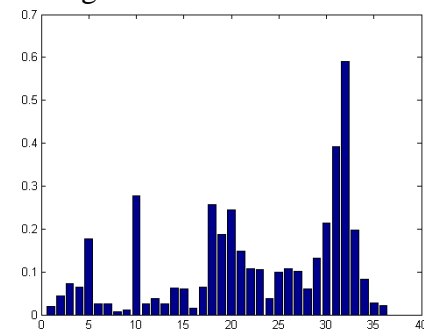
Semen



Nitik

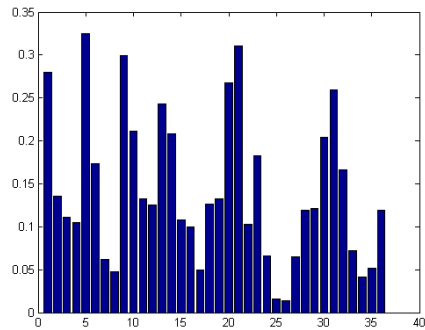


Background



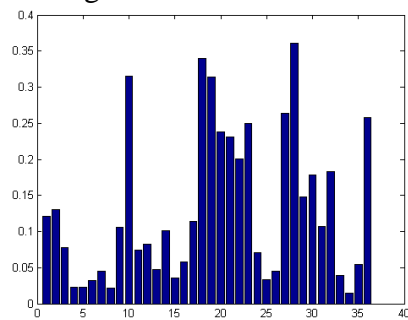
HOG model klasifikasi motif lunglungan

Lunglungan (Objek)

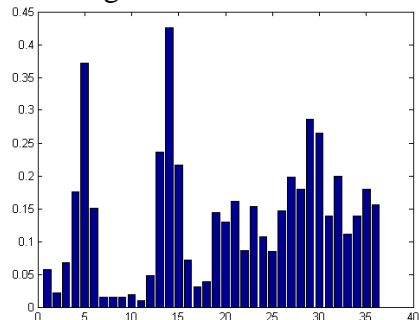


Non Lunglungan (Background)

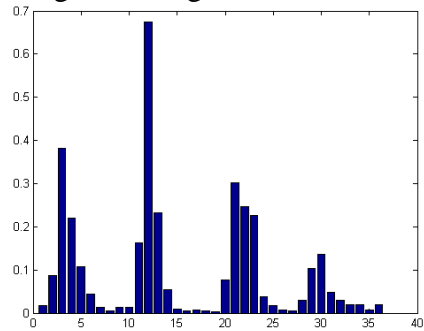
Parang



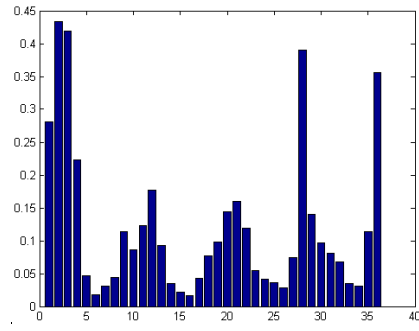
Kawung



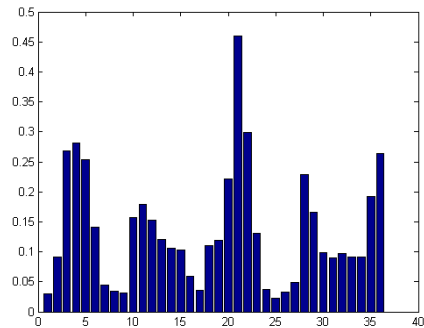
Megemendung



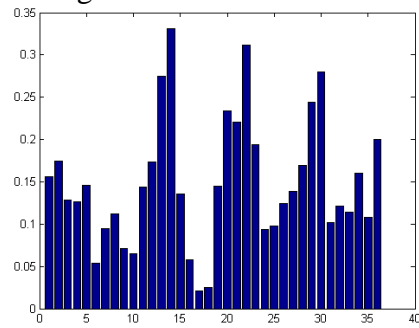
Semen



Nitik

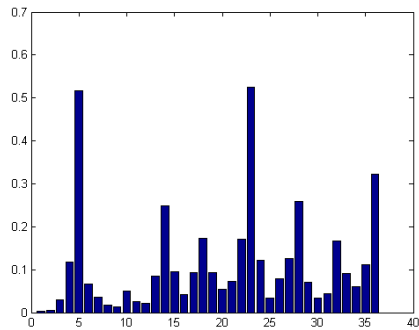


Background



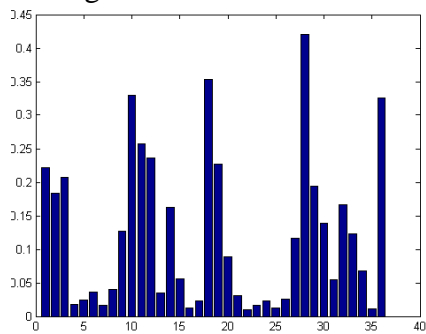
HOG model klasifikasi motif megamendung

Megamendung (Objek)

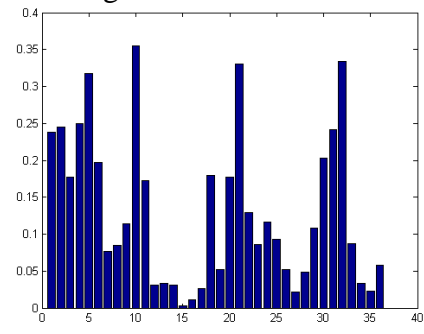


Non Megamendung (Background)

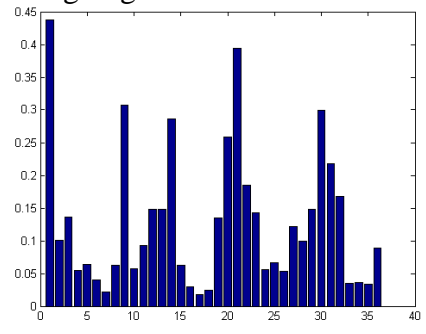
Parang



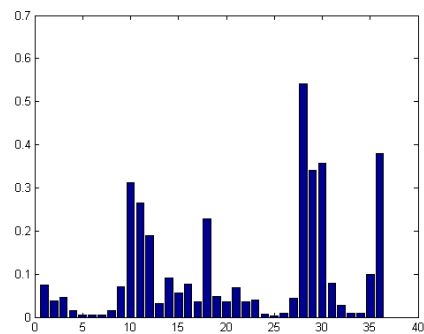
Kawung



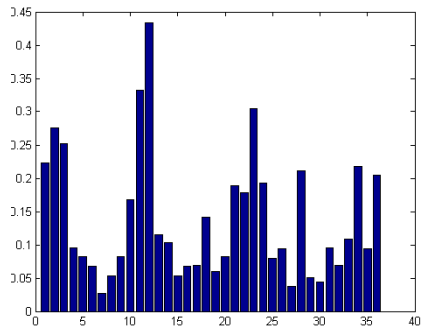
Lunglungan



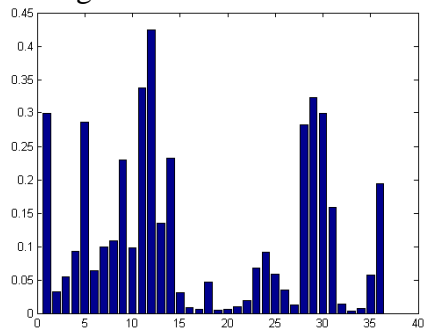
Semen



Nitik

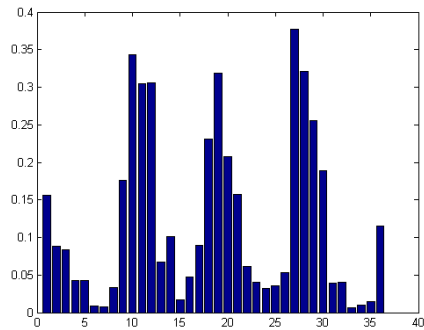


Background



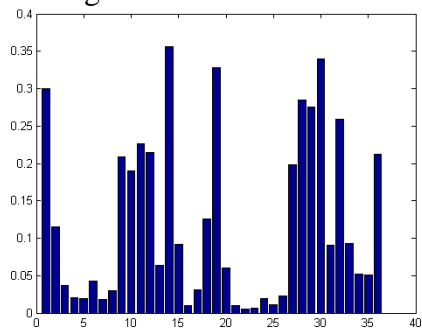
Model klasifikasi motif semen

Semen (Objek)

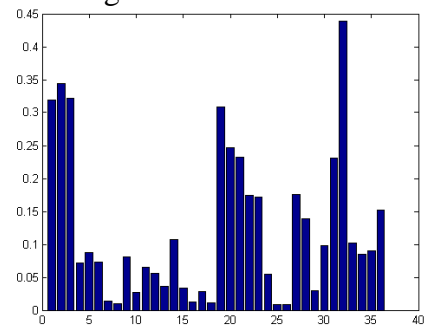


Non Semen (Background)

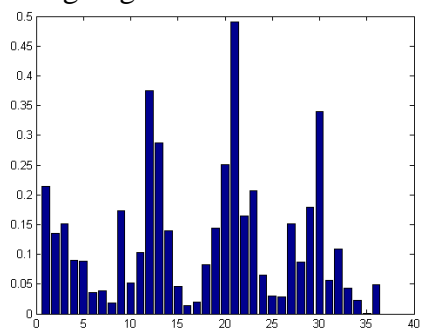
Parang



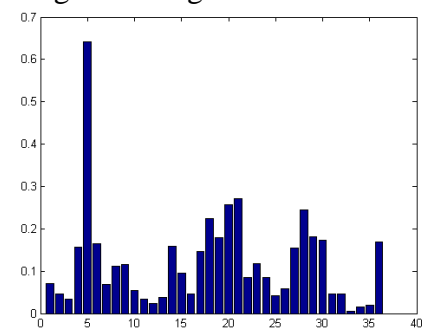
Kawung



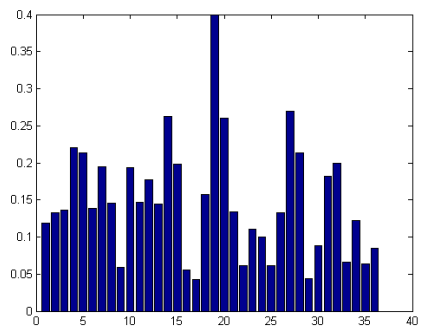
Lungungan



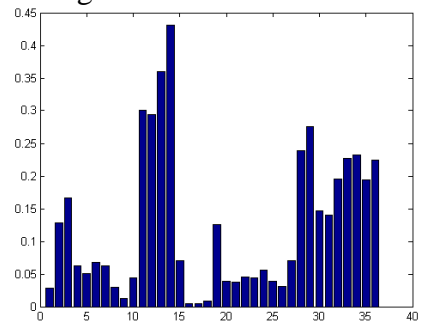
Megamendung



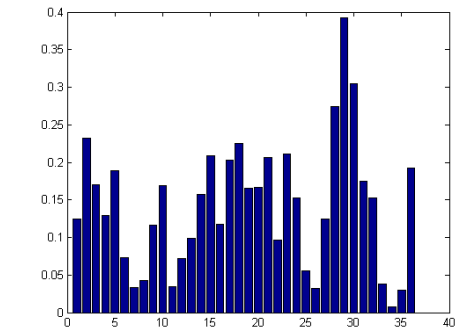
Nitik



Background

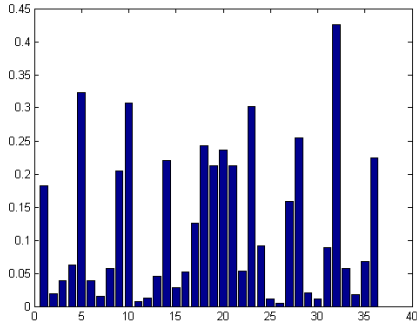


Model klasifikasi motif nitik
Nitik (Objek)

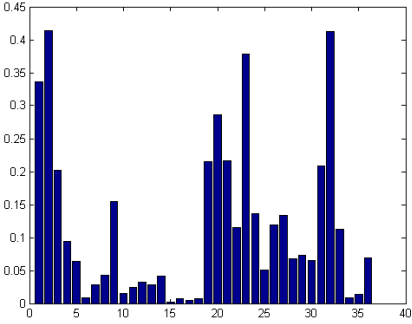


Non Nitik (Background)

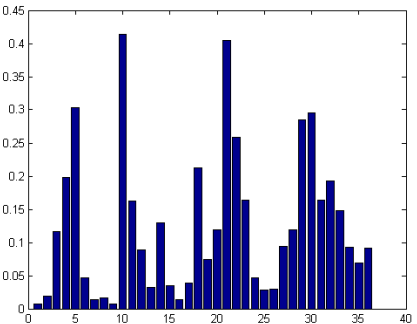
Parang



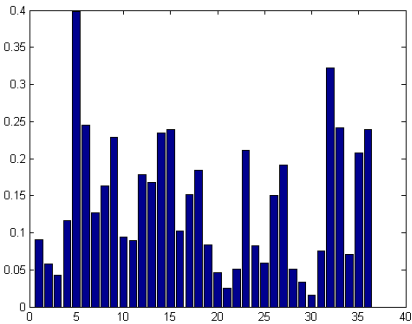
Kawung



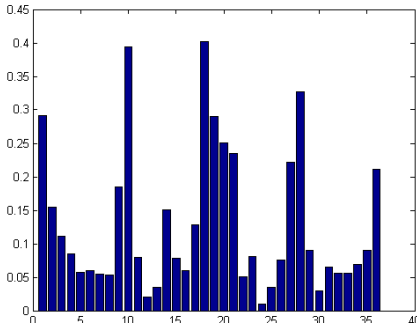
Lunglungan



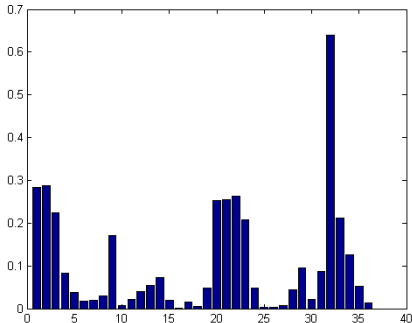
Megamendung



Semen



Background



Biografi Penulis



Penulis, M. Nur Fuad, lahir di Kota Blitar pada tanggal 22 Maret 1992. Penulis adalah anak pertama dari tiga bersaudara. Penulis menghabiskan masa kanak-kanak di Kota Blitar dan menjalani masa remaja di Kabupaten Kediri.

Penulis menempuh pendidikan formal di MI Perwanida, Blitar (1998-2004), SMPN 1 Mojo, Kediri (2004-2007), dan SMAN 1 Mojo, Kediri (2007-2010). Pada tahun 2010-2015, penulis melanjutkan pendidikan S1 di Jurusan Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang, Jawa Timur. Pada tahun 2015-2017, penulis melanjutkan pendidikan Magister S2 pada Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya, Jawa Timur.

Penulis mengambil bidang minat Komputasi Cerdas dan Visualisasi pada jurusan Teknik Informatika. Selama dalam pendidikan S1, penulis pernah menjadi asisten laboratorium yang mengajar kelas praktikum Dasar-Dasar Pemrograman, Pemrograman Berorientasi Objek dan Sistem Terdistribusi. Penulis dapat dihubungi melalui alamat email nurfuadmail@gmail.com